Resource and data management service prototype

Deliverable D2.4

FFG – IKT der Zukunft
SHAPE Project
2014 – 845638

Table 1 Document Information

Project acronym:	SHAPE
Project full title:	Safety-critical Human- & dAta-centric Process management in Engineering projects
Work package:	2
Document number:	2.4
Document title:	Resource and data management service prototype
Version:	1
Delivery date:	01 October 2016 (M4)
Actual publication date:	——————
Dissemination level:	Public
Nature:	Report
Editor(s) / lead beneficiary:	WU Vienna
Author(s):	Giray Havur, Cristina Cabanillas, Saimir Bala, Simon Steyskal, Jan Mendling, Axel Polleres
Reviewer(s):	Axel Polleres, Simon Steyskal

Contents

1 Executive Summary

This document is part of work package 2 (WP2) of the SHAPE project[1]. It reports work performed under Task 2.4 *Resource and data management service prototype* of Work Package 2: *Semantic Models for Mining & Monitoring Process Relevant Data*.

We have described the developed semantic models [1], the reasoning techniques [2, 3], the mining techniques [4–6], the process adaptation techniques [7, 8], and the proposed architecture architecture of the framework [9] in the previous deliverables. In this deliverable, we focus on the implementation of our framework prototype for process management in complex engineering projects as an extension of a well-known BPMS. The framework components are built on the work described in the previous deliverables.

The structure of this deliverable is as follows: In Section 2, we further describe the need for our safety-critical human- and data- centric process management framework. In Section 3, we explore the challenges of the processes in engineering projects which are subject to a large amount of regulations and restrictions, i.e. temporal, resource-related and logistical restrictions, as described in the industry scenario. In Section 4 and 5, we describe the implementation of our prototype in detail. In Section 6, we provide a literature survey on similar work.

Regarding publications, this deliverable has yielded the following conference papers:

- Giray Havur, Cristina Cabanillas, Jan Mendling, and Axel Polleres. *Resource Allocation with Dependencies in Business Process Management Systems.* In BPM Forum 2016.
- Saimir Bala, Cristina Cabanillas, Alois Haselböck, Giray Havur, Jan Mendling, Axel Polleres, Simon Sperl, and Simon Steyskal. A Framework for Safety-critical Process Management in Engineering Projects. In SIMPDA 2015.
- Saimir Bala, Giray Havur, Simon Sperl, Simon Steyskal, Alois Haselböck, Jan Mendling, and Axel Polleres. SHAPEworks: A BPMS Extension for Complex Process Management. In BPM Demo 2016.

2 Introduction

Deployments of technical infrastructure products are a crucial part in the value-creation chain of production systems for large-scale infrastructure providers. Examples of a large-scale, complex engineering process in a distributed and heterogeneous

[1] https://ai.wu.ac.at/shape-project/

environment are the construction of a railway system comprising electronic interlocking systems, European train control systems, operator terminals, railroad crossing systems, etc.; all of the systems are available in different versions using a variety of technologies. It is often necessary to offer, customize, integrate, and deliver a subset of these components for a particular customer project, e.g. the equipment of several train stations with an electronic switching unit in combination with a train control system based on radio communication for a local railway company. Configurators are engineering tools for planning and customizing a product. Each subsystem comes with its own, specialized engineering and verification tools. Therefore, configuring and combining these subsystem data to a coherent and consistent system has to follow a complex, collaborative process.

A challenge is the management and monitoring of such complex, yet mostly informally described, engineering processes that involve loosely integrated components, configurators and software systems [10]. Nowadays, many of the steps required (e.g. resource scheduling, document generation, compliance checking) tend to be done manually, which is error-prone and leads to high process execution times, hence potentially affecting costs in a negative way.

In this deliverable, we explore this domain and present a service prototype for process management in complex engineering processes that includes the formalization of human-centric process models, the integration of heterogeneous data sources, rule enforcement and compliance checking automation, and adaptability, among others. Furthermore, we describe solutions to support the functionalities required by every framework component as well as a proof-of-concept implementation of the framework that can be integrated with the business process management system Camunda. The goal is to help to develop ICT support for more rigorous and verifiable process management.

3 Motivation

In the following, we describe an industry scenario that exemplifies the characteristics of complex engineering processes and define a set of system requirements for the challenges identified in it.

3.1 Industry Scenario

Activities to create complete, valid and reliable planning, and customization process data for a product deployment are part of an overarching engineering process that is of crucial importance for the success of a project in a distributed, heterogeneous

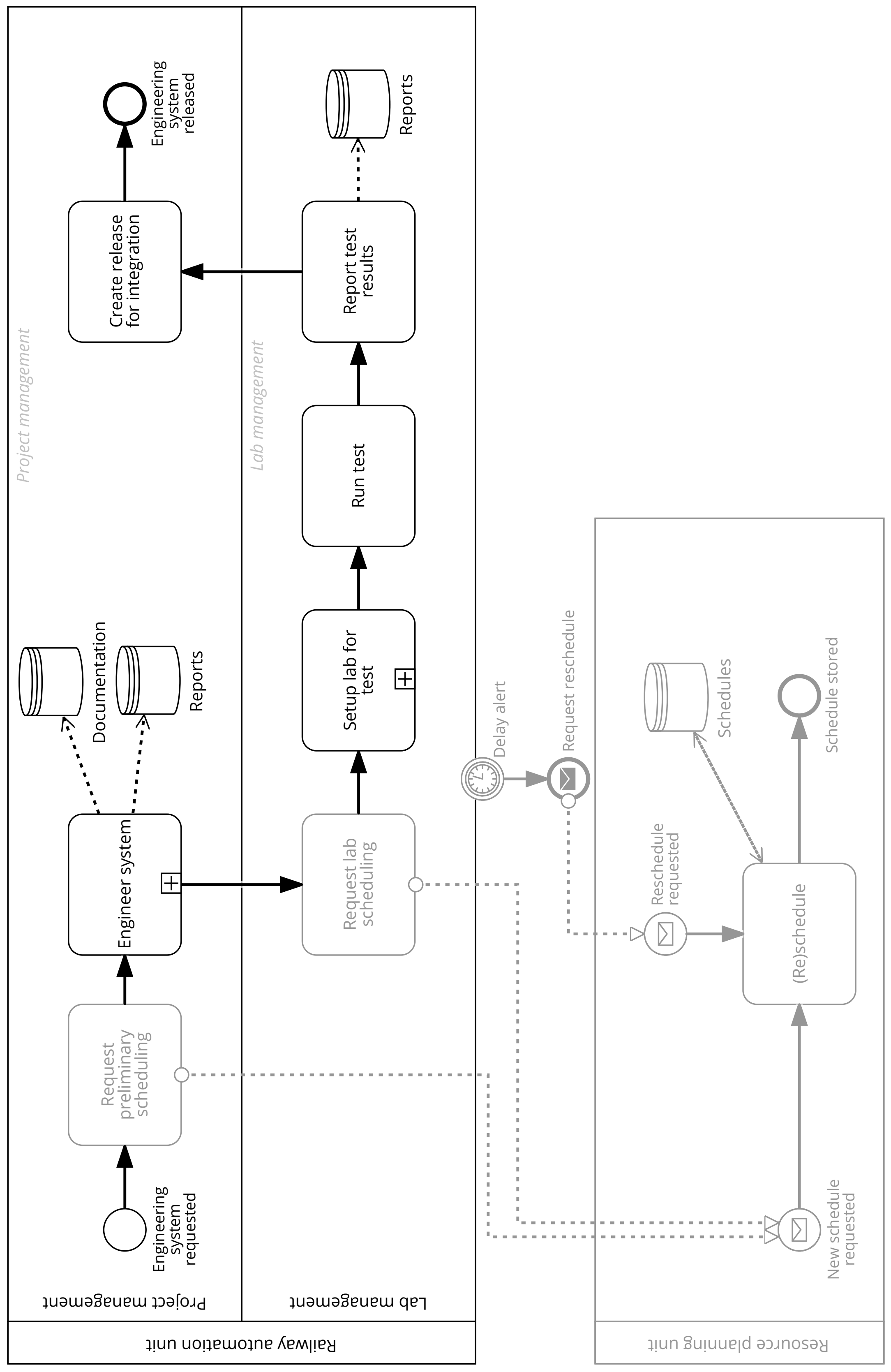


Figure 1 Generic engineering process in the railway automation domain

environment. Fig. 1 depicts a generic engineering process for building a new infrastructure system in the railway automation domain modeled with Business Process Model and Notation (BPMN) [11].

The engineering process itself is represented in the pool *Railway automation unit* and comprises the building and testing of the system. The pool *Resource planning unit* as well as the activities depicted in gray represent a meta-process comprising scheduling activities that are performed in the background in order to enable the completion of the engineering process in compliance with a set of restrictions (temporal and logistics, among others) while making an appropriate use of the resources available. Resource allocation is of great importance to large-scale engineering processes in which a large variety of different resources, ranging from laboratories and specific hardware to engineers responsible for the correct execution of the process, are involved and unexpected situations may have critical consequences (e.g., delays resulting in unplanned higher costs).

Hence, the first step consists of scheduling the building of the system. Building the system is, in turn, a process composed of several activities (potentially operating on different levels of abstraction) each involving a large variety of different resources, data sources, and data formats used. Specifically, the customer provides input data in form of, e.g., XML documents representing railway topology plans, signal and route tables, etc., which are used by the engineers to configure the product. Typically, several configuration tools are involved in that process too, complemented by version control and documentation activities. The result is a set of data of various kinds and formats (i.e., XML, JSON, and alike) such as bill of material (BOM), assembly plans, software configuration parameters, and all other documents and information required for the testing, integration, and installation of the system. Additionally, we map all gathered data to a common extendable RDF model in order to make use of standard data integration and processing strategies from the Semantic Web (e.g., OWL, SPARQL, SHACL, etc.). The engineering project manager orchestrates and monitors these engineering tasks. Besides, further data is generated during the execution of the subprocess *Engineer system* in the form of, e.g., emails exchanged between the process participants.

Once the system is built, it must be tested before it is released for its use. That procedure takes place in laboratories and comprises two phases: the test setup and run phases. Like before, it is necessary to schedule these activities taking into consideration the setting and all the restrictions for the execution of the activities. The setting is the following: there are several space units distributed into two laboratories and several units of different types of hardware for conducting the testing. The employees of the organization involved in these activities are specialized in the execution of specific testing phases for specific types of systems, i.e. there may be an engineer who can perform the setup for a system S_1 and the test execution for a system S_2, another engineer who cannot be involved in the testing of system S_1 but can perform the two activities for the system S_2, and so on. As for the restrictions, they are as follows: each task involved in these two phases requires a specific set of resources for its completion. In particular, the setup tasks usually require one employee working on one unit of a specific type of hardware in a laboratory, and the run activity usually requires several employees for its execution. Besides, a test can only be executed if the whole setup takes place in the same laboratory. In addition, for the scheduling it is necessary to take into account that other instances of the same or different processes might be under execution at the same time and they might share resources.

The setup and the run test activities will then be executed according to the plan. Similar to the engineering step, data comprising the results of the tests, emails, Version Control System (VCS) file updates and the like, is generated during the testing steps. Railway projects also generate other types of data which play a role in the running system, e.g., cut plans, signals form the tracks, actual user data of the system, etc. The latter can be useful when it comes to monitoring safety-critical processes during their execution. Given the great number of software tools involved in the process, our scenario focuses more on the software engineering aspect of the railway domain. Hence, we use VCS logs to track the evolution of the artifacts that are produced during such software engineering process. Nevertheless, it can be extended towards all kinds of artifacts that are stored in VCSs, such as the different versions of outputs from engineering tools.

When the testing of the system is finished, a final report is written and archived with the information generated containing the description of the test cases, test data, test results, and the outline of the findings. Responsible for the final version of this report is the testing project manager. Finally, the engineering project manager deploys a complete and tested version of the engineering system and the integration team takes over the installation of the product.

Note that unexpected situations may cause delays in the completion of any of the activities involved in the engineering process. It is important to detect such delays as soon as possible in order to properly schedule the use of resources and figure out when the process can be finished under the new circumstances. Therefore, rescheduling may be required at any point, involving all the aforementioned restrictions and possibly new ones.

3.2 Challenges and Technical Requirements

A number of issues are involved in the industry scenario previously described when it comes to automating its execution. From the analysis of the process description, the following challenges have been identified:

Challenge 1: Integrated description of processes, constraints, resources and data. Operating with processes like the one described before implies taking not only the order of execution of the process activities and behavioural constraints typically enforced in the process model into consideration, but also information related to other business processes perspectives, such as resources and data requirements, as well as regulations affecting, e.g., the use of these. Several formal languages are at hand for describing processes [11], constraints [12], resources [13] and data (e.g. XML) separately. However, a challenge is to define all them in an integrated manner with a model that provides rich querying capabilities to support analysis automation, status monitoring or respectively, the verification of constraints and consistency.

Therefore, a system for automating processes like the engineering process would require *an integrated semantic model to describe and monitor processes, resources, constraints and data (RQ1).*

Challenge 2: Integration and monitoring of structured and unstructured data. To a high degree, engineering steps are the input for state changes of the process, often only visible as manipulation of data. Hence, a engineering process must also incorporate these data smoothly for monitoring control flow, version updates, data storage, and email notification. To this end, various types of systems have to be integrated including their structured (e.g., logs from tools, or databases) and unstructured data (e.g., by mail traffic, or ticketing systems). Up until now, these data are hardly integrated and are monitored mostly manually.

Therefore, techniques to gather relevant information from unstructured or semi-structured data sources, such as emails, VCS repositories and data from tools, and transform them into understandable data structures must be put in place, i.e. a system for automating processes like the engineering process would require *mechanisms to detect and extract structured from unstructured process data (RQ2).*

Challenge 3: Documentation of safety-critical, human and data aspects and compliance checking. Engineering projects have time-critical phases typically prone to sloppy documentation and reduced quality of results. Many of the process steps are required to be documented in prescribed ways by standards and regulations (e.g. SIL [14]). Considerable amount of time is spent in the manual documentation of process steps as well as in the integration of the documentation of separate modules for generating final reports. Furthermore, in such safety-critical environments the use of resources must be optimized and rules must be enforced and their fulfillment ensured. For instance, in our industry scenario we can observe a typical series of data management steps, including: check in a new version of a data file, inform the subsequent data engineer, confirm and document this step in the process engine, etc. The latter two steps could be done automatically once the process engine has detected the check-in into the VCS. This automation would also lead to a significant decrease of the overall execution time as well as a potential reduction in the number errors typically caused by human mistakes.

Therefore, a system for automating processes like the engineering process would require *a method for flexible document generation (RQ3)* as well as *reasoning mechanisms (RQ4)* and *monitoring capabilities (RQ5)* for automating resource allocation and compliance checking.

Challenge 4: Be ready for changes. Despite engineering process definitions are quite stable and might remain unchanged for a long time, an automatic monitoring and a thorough analysis of process models and executions may lead to the discovery of potential improvement points to make processes simpler and less error-prone. Similarly, changes in the schedule of activities and resources can be necessary at any time due to a number of reasons including delays or unexpected unavailability of resources, among others. Currently, all these adaptations require manual work and are prone to errors.

Consequently, methods to detect and deal with changes must be put in place. This might include the monitoring of process executions and the analysis of the generated execution data (e.g. with process mining techniques) to anticipate delays as well as the need of having available mechanisms capable of reallocating the resources according to changeable requirements and circumstances. Therefore, besides requirements RQ4 and RQ5, a system for automating processes like the engineering process would also require *adaptation procedures to react to changing conditions (RQ6).*

Challenge 5: Acceptance and human factors. The overall process management needs to be set up in a non-obtrusive way, such that engineers executing the processes find it useful and easy to use. This is a specific challenge in safety-critical systems, which are developed with a tight timeline. It calls for a design that integrates existing tools and working styles instead of introducing new systems.

Therefore, the automation of processes like the engineering process would require *an integrated system (RQ7)* that provides all the functionality, which involves general features of a Business Process Management System (BPMS) (e.g. process modeling and execution) extended to support the demands of safety-critical, human- and data-centric processes as described in our industry scenario.

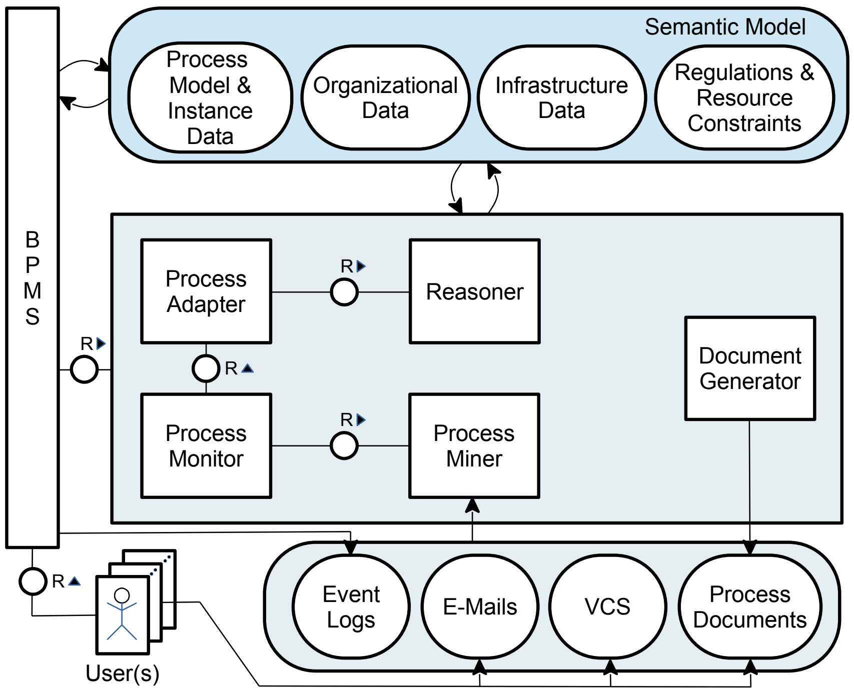


Figure 2 Proposed framework for process management in complex engineering projects

4 Framework

We have designed a framework that provides the support required to address the challenges identified in complex engineering processes. It consists of a data model and five functional modules that interact with a BPMS, as depicted in Fig. 2 using the Fundamental Modeling Concepts (FMC) notation[2].

In order to support *RQ1*, a semantic model encompassing the various types of domain data that must be represented and manipulated must be defined. Hence, this model stores all static and dynamic data used by the BPMS and by the functional components. Specifically, these data include: process models and their instances, organizational data related to human resources, infrastructure data related to non-human resources, and constraints derived from regulations and norms (e.g. SIL) as well as further requirements related to the utilization of resources. The semantic model implicitly operates as a communication channel between the BPMS and all the functional modules and hence, all of them must have read&write access to the model.

Typical functionality of a BPMS include modeling and executing processes. Information about process instances is usually stored in event logs generally including,

[2] http://www.fmc-modeling.org/

among others, temporal and resource information related to the execution of the process activities [15]. In addition to that structured information, as described in Section 3.1, several kinds of unstructured and semistructured data are generated during the execution of complex engineering processes, e.g. emails, VCS files and reports. All the data produced during process execution must be analyzed in order to detect anomalies (e.g., deviations from the expected behavior).

The *Process Miner* component of our framework tries to discover as much data relevant to the current state of a process execution as possible, performs the transformations required as specified by *RQ2*, and communicates the information extracted to the *Process Monitor* (*RQ5*) periodically under request. In case the *Process Monitor* reveals a discrepancy between process instance data and the data discovered by the process miner (e.g., a delay), it informs the *Process Adapter* about the discrepancy. The *Process Adapter* analyzes the deviation and responds by proposing an adaptation solution to the BPMS in order to put the process back into a coherent and consistent state, as specified in *RQ6*. The adaptation may consist of small changes that can be performed directly on the BPMS side or, on the contrary, of complex recovery actions that may require reasoning functionalities. In the latter case, the *Reasoner* comes into play by, e.g., doing a new activity or resource scheduling according to the new domain conditions. Therefore, the *Reasoner* can be seen as a supportive component that helps the BPMS with typical activities, such as the scheduling of process activities, and the allocation of resources to those activities in accordance with resource constraints and regulations defined in the semantic model. This covers *RQ4*.

Finally, the *Document Generator* of the framework provides support for *RQ3* by helping to fill out the documents that must be generated as output of process activities. As mentioned before, this automation is expected to decrease reporting errors, especially in documents related to auditing.

The design of the framework as an extension of the functionality present in existing BPMSs attends to *RQ7* and hence, it intends to increase the acceptance by users familiar with Business Process Management (BPM).

In the following, we describe our solution for the implementation of the functionality provided by the most domain specific components of the framework.

4.1 Semantic Model

Aiming at automation, we believe that Semantic Web technologies provide the most appropriate means for (i) integrating and representing domain-specific (heterogeneous) knowledge in a consistent and coherent format, and (ii) querying and processing integrated knowledge.

Therefore, following the METHONTOLOGY approach [16], we have developed an engineering domain ontology [17] that integrated three different domains of interest relevant for our approach, namely: (i) engineering domain and organizational (i.e. resource-related) knowledge; (ii) business processes; and (iii) regulations and policies [18].

4.1.1 Representing Infrastructural & Organizational Knowledge

One of the first steps for developing an ontology according to the METHONTOLOGY approach involves the definition of an *Ontology Requirements Specification Document* [19]. In order to address the requirements gathered throughout that process we decided to adopt parts of the organizational meta model described in [20] and enriched it with concepts for modeling teams [21] (cf. Fig. 3) for representing infrastructural & organizational knowledge. Using these two meta models presents an advantage. Specifically, the organizational meta model described in [20] has been used to design a language for defining resource assignment conditions in process models called Resource Assignment Language (RAL) [13]. As was shown in [22], that language can be seamlessly integrated in existing process modeling notations, such as BPMN, thus enriching the process models with expressive resource assignments that cover a variety of needs described by the creation patterns of the well-known workflow resource patterns [20]. Furthermore, a graphical notation was later designed with the same expressive power as RAL in order to help the modeler to define resource assignments in process models [23]. The meta model for teamwork assignment was also considered to develop an extension of RAL called RALTeam [21], which, however, lacks a graphical notation so far. Therefore, if support for these expressive notations were introduced in the BPMS, the ontology would support them at the same time as it supports less expressive means of assigning resources to process activities (e.g. based on organizational positions).

4.1.2 Representing Business Processes

Driven by the requirements of our resource allocation approach (cf. Section 4.2.1), we decided to transform BPMN models into timed Petri nets [24] as an intermediary format for reasoning tasks (e.g., scheduling [25]) by using the transformation proposed in [26, 27], and store these Petri nets in our ontology. Note that the user only interacts with the BPMN model while using the system as we use the timed Petri net

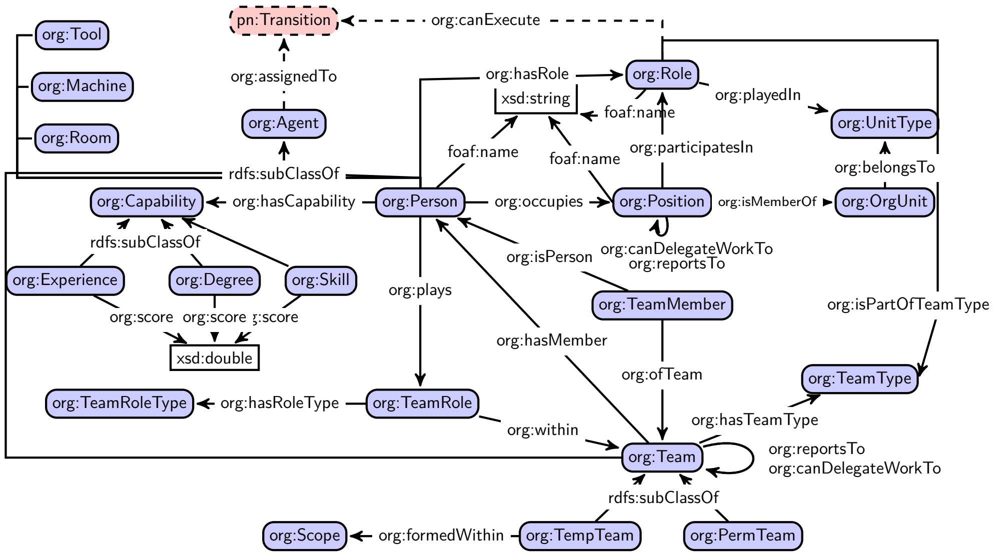


Figure 3 Ontology for infrastructural & organizational knowledge.

representation internally. There are several reasons for using Petri nets for process modeling [28], namely:

- *Clear and precise definition:* Semantics of the classical Petri net is defined formally.
- *Expressiveness:* The primitives needed to model a business process (e.g. routing constructs, choices, etc.) are supported.
- *Tool-independent:* Petri nets have mappings to/from different business modelling standards [29]. Moreover, this immunizes our ontology from changes in business modeling standards.

For modeling Petri nets themselves we adopted selected concepts of the Petri Net Markup Language (PNML) [30] and represented them in terms of an RDFS ontology (cf. Fig. 4).

4.1.3 Extracting and Specifying Compliance Rules

One of the most important aspects of dealing with safety-critical human- and data-centric processes is providing means for proving that business processes comply with relevant guidelines such as domain-specific norms and regulations, or workflow patterns. As illustrated in Fig. 5 and described in [31], establishing proper compliance checking functionalities typically requires to extract and interpret a set of *Compliance Objectives* from respective *Compliance Requirements* first, before those objectives are specified in terms of *Compliance Rules/Constraints* using an appropriate specification

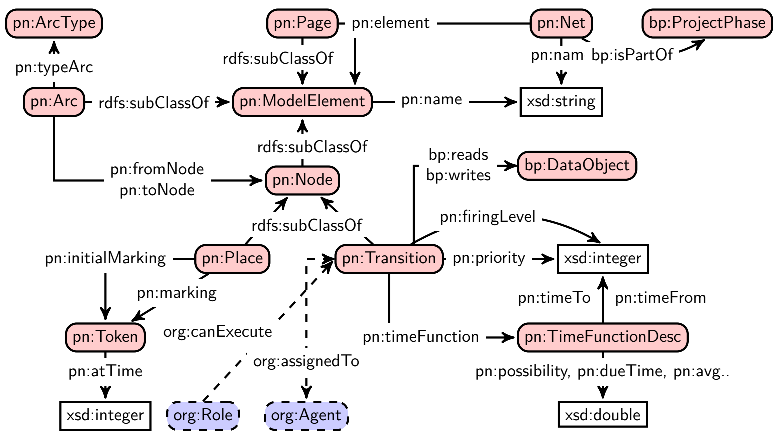


Figure 4 Ontology for representing processes and process instances.

language (i.e. a language capable of representing all types of compliance rules/constraints relevant for the respective domain of interest). Specified compliance rules and constraints are then subsequently used by a monitoring/compliance checking engine for verifying correct and valid execution of business processes w.r.t. previously defined rules.

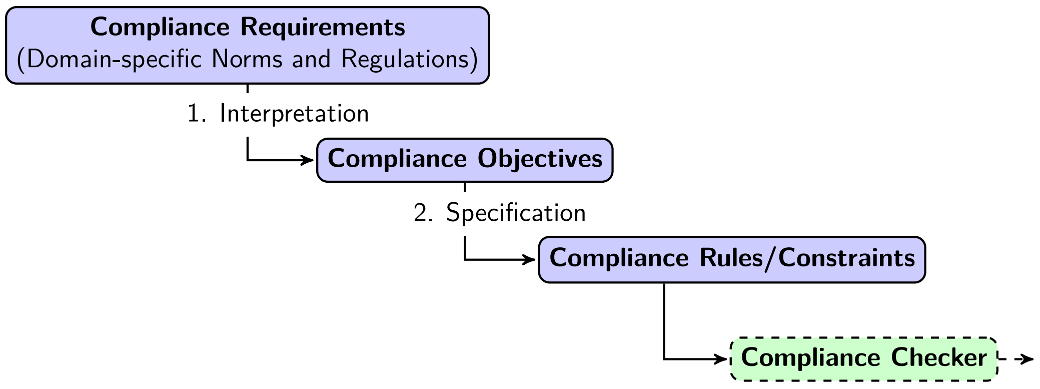


Figure 5 General approach for business process compliance monitoring [31].

1. *Identifying Compliance Objectives:* Organizations have to deal with an increasing number of norms and regulations that stem from various compliance sources, such normative laws and requirements. In the railway domain processes have to be compliant with specific European Norms (i.e. 50126, 50128, and 50129). For example, EN50126 defines guidelines for managing Reliability, Availability, Maintainability, and Safety (RAMS) of safety-critical business processes, where we extracted objectives, requirements, deliverables, and validation activities defined in all phases of the RAMS lifecycle [1, 32].
2. *Representing Compliance Rules and Constraints:* Since all process relevant data

are stored in RDF, we plan to utilize recent advancements in the area of constraint checking for RDF, i.e. the Shapes Constraint Language (SHACL) [33] for representing and validating identified compliance objectives. Since constraints expressed in SHACL are internally mapped to corresponding SPARQL queries, we can further complement our own compliance constraints with already existing approaches for compliance checking using SPARQL such as [34]. Compliance constraints expressed in SHACL are tightly integrated with the underlying ontology and can be validated during both design time and runtime[3].

4.2 Reasoner

The *reasoner* module supports our framework on top of the engineering domain ontology in two folds: (i) by performing automated resource allocation described in the declarative formalism Answer Set Programming (ASP) [35], and (ii) by querying the ontology for compliance checking. We also looked at other declarative programming paradigms (e.g., CLP(FD)), and our initial findings confirm the advantages of using ASP [36,37]. Some of these advantages are as follows:

- Compact, declarative and intuitive problem encoding
- Rapid prototyping and easy maintenance (e.g., no need to define heuristics)
- Complex reasoning modes (e.g., weight optimization)
- Ability to model effectively incomplete specifications
- Efficient solvers (e.g., *clingo*)

In the literature, ASP is preferable when the size of the problem does not explode the grounding of the program [37,38]. We show that our resource allocation encoding in ASP is applicable to the problems of business processes at a real-world scale [25].

4.2.1 Resource Allocation

Resource allocation aims at scheduling activities of a business process and properly distributing available resources among scheduled activities. We address the problem of allocating the resources available to the activities in the running process instances in a time optimal way, i.e. process instances are completed in the minimum amount of

[3] For a more detailed introduction on utilizing SHACL for defining custom constraints, we refer the interested reader to [33].

time. Therefore, our resource allocation technique makes business process executions effective and efficient.

We encode the resource allocation problem in Answer Set Programming (ASP) [35], a declarative (logic programming style) paradigm. Its expressive representation language, efficient solvers, and ease of use facilitate implementation of combinatorial search and optimization problems (primarily *NP-hard*) such as resource allocation. Therefore, modifying, refining, and extending our resource allocation encoding is uncomplicated due to the strong declarative aspect of ASP. We use the ASP solver *clasp* [35] for our purpose as it has proved to be one of the most efficient implementations available [39]. Another complex reasoning extension supported in *clasp* are weight optimization statements [35] to indicate preferences between possible answer sets.

Resources are defined in the engineering domain ontology (the organizational data and the infrastructure data) where they are characterized by a *type* and can have one or more *attributes*. In particular, any resource type (e.g., org:Person in Fig. 3) is a subclass of org:Agent. The attributes are all of type rdf:Property. The organizational data consists of human resources, their attributes (e.g. their name, role(s), experience level, etc.) and current availabilities stored in the ontology. In the same fashion, infrastructure data represents material resources (i.e. tools, machines, rooms) and their availabilities. Resource allocation considers resources to be *discrete* and *cumulative*. Discrete resources are either fully available or fully busy/occupied. This applies to many types of resources, e.g. people, software or hardware. However, for certain types of infrastructure, availability can be partial at a specific point in time. For instance, a room's occupancy changes over time. Such a cumulative resource is hence characterized by its *dynamic* attribute (available space in the room) and it can be allocated to more than one activity at a time. Any statement in our ontology can be easily incorporated as the input of our problem encoding [40]. The following example shows an excerpt of organizational data in the ontology and its equivalent in ASP.

```
# Organizational data
:glen a org:Person; foaf:name "Glen";
      org:occupies testeng.
:testeng a org:Position; foaf:name "Test Engineer";
         org:participatesIn labmng.
:labmng a org:Role; foaf:name "Lab Management".

% Equivalent ASP encoding
person(glen). name(glen,"Glen").
occupies(glen,testeng).
position(testeng). name(testeng,"Test Engineer").
```

```
participatesIn(testeng,labmng).
role(labmng). name(labmng,"Lab Management").
```

There are two main operations under resource allocation: *Allocation of resources* and *re-allocation of resources as adaptation*.

Allocation of resources deals with the assignment of resources and time intervals to the execution of process activities. It can be seen as a two-step definition of restrictions. First, the so-called *resource assignments* must be defined, i.e., the restrictions that determine which resources can be involved in the activities [13] according to their properties. The outcome of resource assignment is one or more *resource sets* with the set of resources that can be potentially allocated to an activity at run time. The second step assigns cardinality to the resource sets such that different settings can be described.

As mentioned in Section 4.1, there exist languages for assigning resource sets to process activities [13, 41–43]. However, cardinality is generally disregarded under the assumption that only one resource will be allocated to each process activity. This is a limitation of current BPMS, which we overcome in our proposed framework.

The main temporal aspect is determined by the expected duration of the activities. The duration can be predefined according to the type of activity or calculated from previous executions, usually taking the average duration as reference. This information can be included in the executable process model as a property of an activity (e.g. with BPMN [11]) or can be modelled externally. As for the variable activity durations depending of the resource allocation, three specificity levels can be distinguished:

- *Role-based duration*, i.e., a triple $(activity, role, duration)$ stating the (minimum/average) amount of time that it takes to the resources within a specific resource set (i.e., cardinality is disregarded) to execute instances of a certain activity.
- *Resource-based duration*, i.e., a triple $(activity, resource, duration)$ stating the (minimum/average) amount of time that it takes to a concrete resource to execute instances of a certain activity.
- *Aggregation-based duration*, i.e., a triple $(activity, group, duration)$ stating the (minimum/average) amount of time that it takes to a specific group to execute instances of a certain activity. In this report, we use *group* to refer to a set of human resources that work together in the completion of a work item, i.e., cardinality is considered. Therefore, a *group* might be composed of resources from different roles which may not necessarily share a specific role-based duration. An aggregation function must be implemented in order to derive the most appropriate

duration for an activity when a group is allocated to it. The definition of that function is up to the organization.

Given (i) a process model and its instance data; (ii) organizational and infrastructure data; (iii) resource requirements, i.e. the characteristics of the resources that are involved in each activity to be allocated (e.g. roles or skills); (iv) temporal requirements; and (v) regulations such as access-control constraints [13], i.e. *separation of duties (SoD)* and *binding of duties (BoD)*, the ASP solver finds an optimal allocation. The aforementioned functionalities and the entire associated ASP encoding are detailed in [9].

While executing the process instance, changes may be introduced to input used for allocation. For instance, organizational data may change in case of absence, regulations may be modified or simply execution of activities may delay. In some cases, such a change in the ontology directly affects a running process instance, and therefore, the process monitor informs the process adapter. The process adapter may decide that the allocation should be performed again. *Adaptive re-allocation* is a key functionality in this scenario and it is indispensable for safety-critical, human- and data- centric process management.

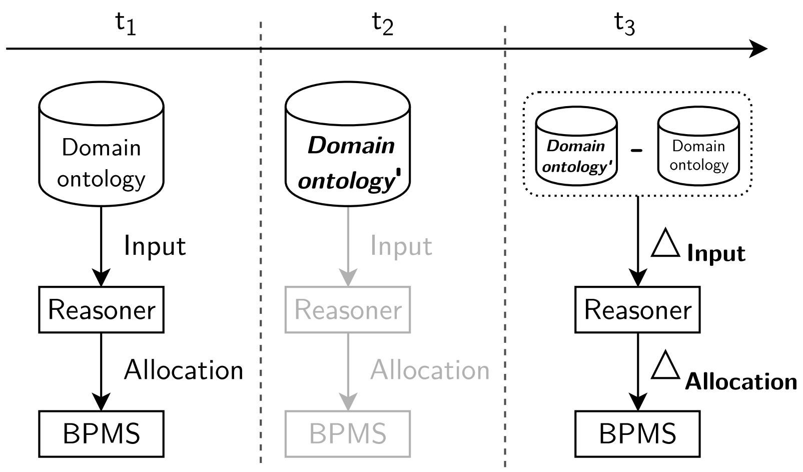


Figure 6 Adaptive re-allocation timeline

Fig. 6 shows this scenario in three consecutive time steps: After allocating resources to a process instance at t_1, some changes are introduced at t_2 that interfere with the original allocation, and hence, an adaptive reallocation is performed at t_3. The reasoner computes a delta allocation, i.e. the original allocation is preserved as much as possible. Therefore, some activities might be rescheduled, and others might be shifted and/or reallocated to some different resources in a *minimal* fashion.

4.2.2 Compliance Checking

As mentioned previously, we define compliance constraints over business processes using SHACL. In order to do so, we translate each compliance objective to a corresponding SPARQL query first, before embedding it in a respective constraint component, which itself can then be integrated into the ontology [1].

4.3 Process Monitor and Process Adapter

Changes and deviations to the processes may occur during execution. For instance, new rules and regulations may require the process to operate differently. We want our framework to be able to handle these unexpected events.

The process data and the evidence from the process miner are compared by the process monitor for detecting deviations. The main idea is that the process adapter is informed about the deviations, therefore it minimizes the impact of these deviations in the running process instances by offering a recovery strategy. The process monitor and process adapter address the requirements *RQ5* and *RQ6*, respectively. The process monitor is able to run several algorithms for monitoring both the process behaviour and the process compliance to rules and regulations. This is performed by checking the current process constraints against the data from our semantic model.

The process adapter is in charge of handling exceptions that arise from the process monitor. This component acts in two different ways: Either it *i)* corrects process behaviour with minimal intervention, or, in case a more complex adaptation is required, *ii)* it stops the process and notifies the reasoner for planning an adaptation.

4.4 Process Miner

Traditional process mining algorithms [15, 44] are able to give valuable insights into the different perspectives of a business process. Process models inferred from log files can further be analyzed for bottlenecks, performance, deviations from the expected behaviour, compliance with rules and regulations, etc. Regardless of the perspective they aim at mining as well as the type of process modeling notation used to represent the outcome (declarative versus imperative process mining), all these process mining algorithms require properly structured data. Specifically, they must comply with the XES [45] meta model. Any of the existing process mining techniques is a candidate to be used for the implementation of the *Process Miner* component in regard to the functionality related to traditional process mining and the decision should be made according to the specific characteristics desired.

However, the biggest challenge of this component in our framework is to deal with unstructured and semistructured data generated in the execution of the process activities, generally in the form of VCS files and emails. Although it is hard to mine process models out of such unstructured or semistructured data, some approaches can be used to obtain valuable insights on them. Specifically, [46] allow for transforming semi-structured VCS logs to process activities which can be mined by classic mining algorithms. Poncin et al. [47] developed the FRASR framework, which is enables to answers engineering questions by applying process to software repositories. The challenge here is to identify the relevant events for the files, from a process mining point of view. Di Ciccio et al. [48] propose the MailOfMine approach to discover artful processes laying underneath email collections. Bala et al. [49] adopt a visualization approach by mining project Gantt charts from VCS logs.

Driven by the fact that complex engineering process like the industry scenario described in Section 3.1 are resource-intensive, we have developed a novel approach to extract organizational roles from VCS repositories. VCSs have both structured and unstructured data. On the one hand, they explicitly provide information about the user who performed changes in some file(s) and the time at which she committed the new version(s). On the other hand, they have a textual part typically carrying a comment that explains the changes performed on the file(s). Note that these kind of data are similar to data from email. In fact, both emails and git comments have in common information about the user, the timestamp, and a textual description. Moreover, we use an ontology (cf. Section 4.1) to store mining insights. That is, we allow for the integration of all types of data that can be represented in RDF, including data coming from engineering tools. There are indeed tools in SVN [50] or Git [51] that allow for sending user commits as emails. Therefore, discovering roles out of such data (and especially when the outcome is combined with the result of mining activities) might help the *Process Monitor* to identify potential deviations regarding the resources that have actually performed specific tasks or manipulated certain information. Hence, it contributes to compliance checking.

Let us see an example of users who use VCS to collaborate on a software development project. Fig. 7 shows a setting with three users named Alice, Bob and Claire. Alice is a development project manager. She works with her colleagues Bob and Claire. Alice is mainly responsible for *feature development*, but she is also involved in *testing* project-management team. Her tasks include the *development* of new features and fixing of related bugs. In her first *commit* she adds a new message where she describes her work. The message to describe her changes is “added new module to demo". In the first row of Table 2, identified by *commit id 1*, Alice’s change is reported. Bob is part of the *testing* project-management team. His task is to ensure that the code submitted by the development team complies with

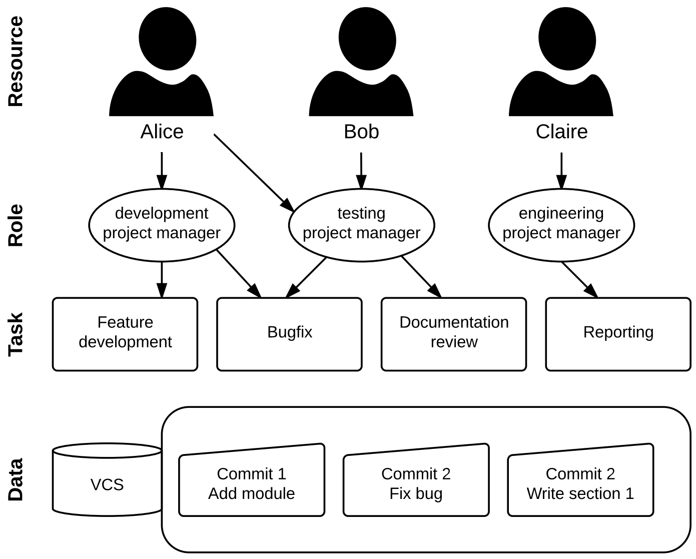


Figure 7 Software project and resources

Id	User	Timestamp	Changed	Comment
1	Alice	2014-10-12 13:29:09	Demo.java rule.txt	Added new module to demo and updated rules
2	Bob	2014-11-01 18:16:52	Setup.exe	Modified the setup interface
3	Alice	2015-06-14 09:13:14	Demo.java	Update the application interface
4	Claire	2015-07-12 15:05:43	graph.svg todo.doc	Define initial process diagram & listed remaining tasks
...	...	...	...	...

Table 2 Example of a VCS log

existing standards and contains no errors. He discovers and fixes some minor bugs in Alice's code and informs Alice on further work needed on the analyzed features. Meanwhile, he commits his changes with *commit id 2* and comments “Modified the setup interface". Consequently, Alice reworks her code and commits a new version as reported in row 3 of Table 2, commenting her work with “Update application interface". As an *engineering project manager*, Claire takes over and starts to work on the documentation. She commits part of her work as in row 4. As the project continues, the work is accordingly stored in the log as shown in the table.

Our approach leverages both on the file types and the comments of the users. We devise an algorithm that classifies users into a set of roles. For that purpose, we approach the role discovery problem as a classification problem. We define two methods: one based on user clustering and one based on commit clustering. A prior step for this is the feature selection. By looking at the commit data, we identify the following features:

- Total number of commits.
- Timeframe between the first and the last commit of a user (i.e. the time he has been working on the project).
- Commit frequency: total number of commits divided by the time frame.
- Commit message length: average number words in the commit comment.
- Keyword count: how often determinate words like “test” or “fix" are used
- Number of files changed.
- Affected file types: how often a file with a certain format (e.g., *.java, *.html) are modified by a user, relative to the total number of modified files

Then, we use the features for two machine learning algorithms. In the first approach we iterate through the users and cluster them using the k-means algorithm. Consequently we build classification models using decision trees. We then train three different datasets individually and cross validate the results. The second approach starts from the commits. The main idea here is that we do not want to assign users to a specific category. Rather, we allow for users having multiple roles and classify their contribution in each commit. We build user profiles that account for fractions of contributions of each user to the different classes. Classes used in the classification for the example described above would be: *Test, Development, Web, Backend, Maintenance, Refactor, Documentation, Design, Build, Data, Tool, Addition, Removal, vcsManagement, Automated, Merge.* Each commit is classified into one of the classes according to its features. Users who committed can be then classified by their commits. The classification can be done both manually and automatically: *i)* rules can be manually inferred by looking for similarities between users with the same role; or *ii)* an automated classification can be performed by using machine learning algorithms. For example, decision trees can be used for an automated classification. In this case the *commit* type percentages are used as features and the manually assigned roles as classes. As a further step, the resulting decision tree models from the different datasets can be cross-validated. The complete approach and its evaluation can be found in [52].

4.5 Document Generator

Safety-critical engineering systems require well-documented process steps. Engineers are in charge of clearly describing their tasks in such a way that it is possible to audit their work. These documents are often manually created. This has at least four drawbacks. First, their creation is laborious. Second, it is error-prone and misaligned in terms of language. Third, it is described at different levels of granularity. Fourth, it is difficult to process and audit afterwards.

A simple example is the following. Engineers need to work on a specific task and use predefined tools. Their tasks and their version tools are specified at the beginning of the project and must be consistent during the project's lifetime. Tool versions must usually be filled in the documentation generated, e.g. in reports. In a big engineering project, tools can be numerous and their versions are far from being user-friendly. This makes the risk of human mistakes very likely.

To assist engineering project managers in producing audible documentations, we have developed a customizable approach for partially automating document generation. In particular, it is able to fill in trivial information (e.g., tool version, user name, task to which the user is assigned, etc) into word processor documents. Our document generation technique is based on templates. These templates consist of evolving documents and are automatically filled in during the workflow, and therefore enable flexible process verification. Our approach generates standard documents which are compatible with predefined word processor programs and can be opened and edited by them.

Document generation comprises four steps, depicted in Figure 8 and explained next. A mapping function is first defined from a process activity to a document which is generated as its outcome. Afterwards, an interpreter is defined which is in charge of filtering the relevant process activities and variables. Process variables are used by the BPMS during the execution of the process. Examples of process variables can be the name of the user that is currently assigned to one activity, the name of the running process instance, and everything that adds data to the executing process in the BPMS. The interpreter is not strictly bound to a particular process nor to a particular template, and is defined externally. This supports changes both in the process and in the template. The writing in the document is triggered by a listener. A listener waits for activity events. As soon as an activity is submitted, the interpreter and the mapping function work together to generate *(variable key,value)* pairs in the document template. This is run iteratively on the document until all the trivial data is filled in.

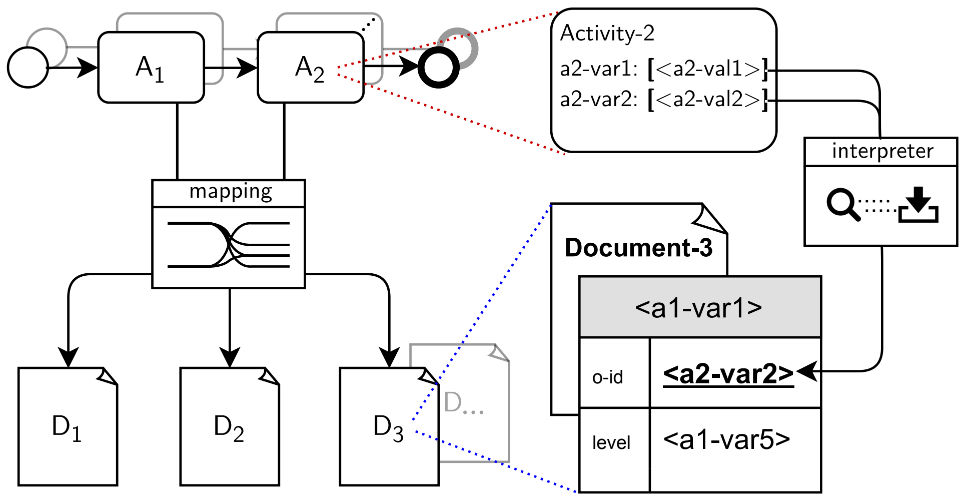


Figure 8 Implementation showing how form values (process variables) from Camunda could look like in a generated document

5 Proof-of-concept Implementation

As a proof of concept we have implemented the main components of the framework discussed in Fig. 2. In this prototype, we aim to bring together functionality from reasoning, process mining and document generation. Fig. 9 shows the software architecture that we use. It considers four main components which interoperate during the execution of a process activity.

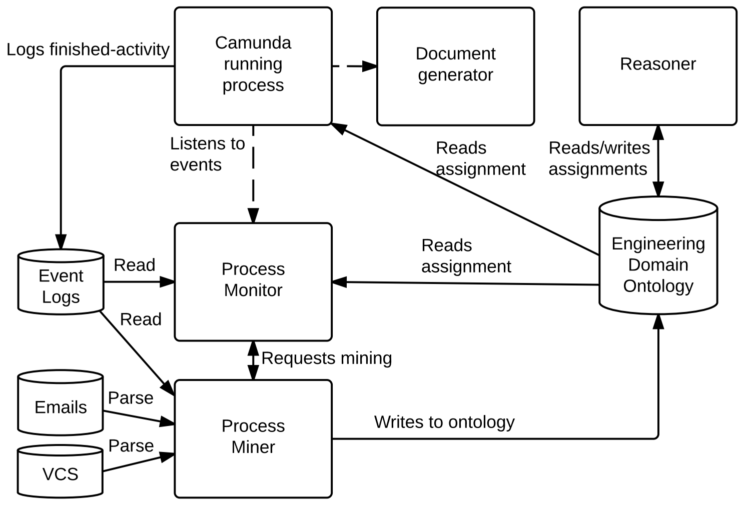


Figure 9 Software architecture of the prototype

Here, we describe the main components of the architecture and their interactions.

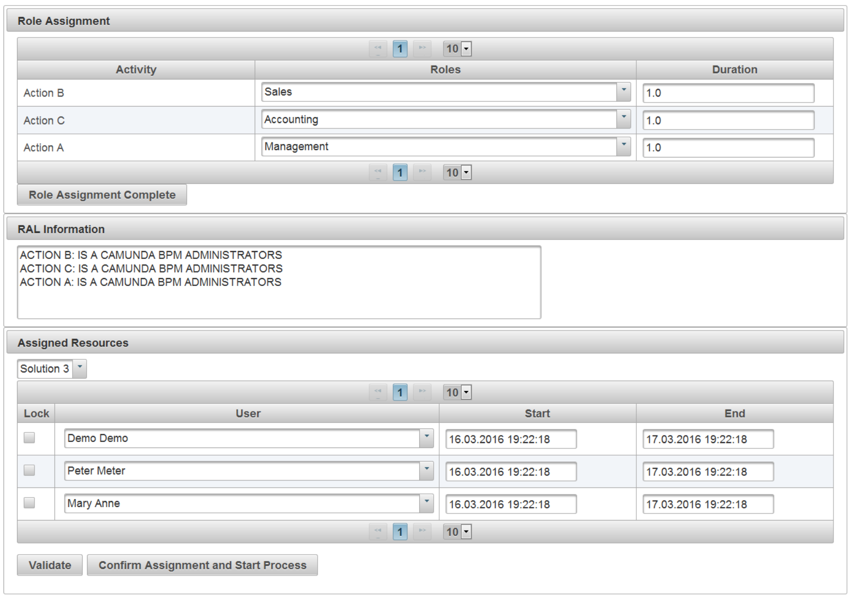
Camunda running process. We use the Camunda BPM engine as our BPMS. Camunda is an open source platform that allows for defining new components and for interacting with its APIs in a custom way. All the process instances that run into Camunda and their data are stored in log files. Camunda uses two main databases to store its logs: *i)* a database for processes that are currently executing; and *ii)* a database for historical information. These two databases can be queried through provided Java or REST APIs. Results are returned as either a set of Plain Old Java Objects(POJOs) or in the JSON format, respectively.

Before an activity starts to run, it first fetches the ontology which contains the set of assignments from existing resources to activities. Consecutively, a resource is assigned to the activity and thus can appear on their task list. When the resources complete their tasks, an event is triggered. This event is listened by the process miner and the document generator components, who can react accordingly. At the same time, the event is stored into the Camunda database of the running instances. Both the running processes database and the history database record similarly-structured data. Furthermore, they can be accessed using the same technology, i.e. the Camunda REST APIs. Hence, we abstract both these databases as a single database in Figure 9 and denote it as *Event Logs*.

Reasoner. The reasoner module is implemented as a Java application connected to the Camunda process engine as an asynchronous service. We use Sesame, an open source framework for creating, parsing, storing, inferencing and querying over our ontology data. With respect to the request, the reasoner either performs resource allocation (cf. Figure 10) by first translating the RDF data into the ASP language, solving the problem instance using the ASP solver *clasp*, and then writing the allocation results back to the triple store; or it validates all contained SHACL constraints and returns potential violation result back to the process engine.

Process Monitor. This component is in charge querying the status of the running processes in Camunda. In case a deviation occurs, for example, a process instance cannot be completed within the assigned schedule, the process monitor must signal out the anomaly. The process adaptation module can use this output to learn the status of the system and subsequently apply an adaptation. This component is implemented as a web client that can read execution logs through the Camunda REST API. Results are returned in the JSON format which are then parsed into POJOs and can be processed by customized monitoring algorithms. In this case the communication happens through periodical queries to the database. An alternative to this is to implement an activity listener that notifies the process monitor whenever a task is completed.

Miner. The miner is in charge of running a number of mining algorithms on the logs



Activity	Roles	Duration
Action B	Sales	1.0
Action C	Accounting	1.0
Action A	Management	1.0

ACTION B: IS A CAMUNDA BPM ADMINISTRATORS
ACTION C: IS A CAMUNDA BPM ADMINISTRATORS
ACTION A: IS A CAMUNDA BPM ADMINISTRATORS

Solution 3

Lock	User	Start	End
	Demo Demo	16.03.2016 19:22:18	17.03.2016 19:22:18
	Peter Meter	16.03.2016 19:22:18	17.03.2016 19:22:18
	Mary Anne	16.03.2016 19:22:18	17.03.2016 19:22:18

Figure 10 Resource allocation interface

from Camunda and from VCSs. Emails and commit messages can also be analysed by using the approaches discussed in [52]. This component is implemented as a web service, which can be called by the process monitor in order to understand how the activities being monitored have performed in the past. Mining algorithms can give new insights into the processes, like for instance actual execution times and several performance indicators of the process. This can contribute to the domain knowledge. Thus, they are stored again into the ontology as RDF.

Document generator. The document generator is in charge of listening to activity submissions and of collecting information from them with the final goal of creating textual documents. This component uses customizable event handlers to process changes of process variables and forms compiled by the users. It is implemented in Java and can be imported as a Java library into several other modules that require document generation from events.

5.1 Maturity

Our prototype is currently a prototype that serves mainly as a proof-of-concept. It shows the advantage of having an integrated solution of different approaches implemented on top of a BPMS. We plan to further develop our extension by adding more features, divided into three levels.

Level 1 (current). Our prototype includes: *i)* resource (re-)allocation with data and resources from Camunda; *ii)* ontology describing the organizational model;

iii) process monitoring that triggers alerts when the process risks running late and the process execution does not respect the allocation; and *iv)* mining history of tasks and resources, and updating ontology with mined data.

Level 2. Our prototype includes: *v)* allocation of non-human resources, which are fully synchronized with the data from the ontology; *vi)* delay prediction by mining history logs; and *vii)* infrastructure model stored in the ontology.

Level 3. Our prototype includes: *viii)* flexible resource (re-)allocation, i.e., allocating resources up to the next decision point of the process, thus enabling a more dynamic schedule; *ix)* mining XOR probabilities, i.e., the likelihood of particular choices made in XOR gates; and *x)* feasibility check using XOR probabilities, i.e., given the likelihood of the path to be executed in the business process and the resources, check if the execution is possible in n amount of time.

5.2 Ongoing development

Scheduling backend. The scheduling program in the reasoning component is updated from an earlier version [25] to current version [53].

SQL console for querying Camunda logs. We are developing a tool for process monitoring. This tool will allow for SQL-like queries on top of Camunda logs. The following snippet exemplifies a possible query on Camunda logs that returns all *userTask* activities that took one month to complete:

```
SELECT ActivityId, ActivityName,
DIFFDATE(month,Activity.START,Activity.END)
AS Duration
FROM Activities
WHERE ActivityType = "userTask"
AND Activity.STATUS = "finished"
ORDER BY Duration DESC
LIMIT 10;
```

To realize this approach, a mapping from Camunda's database schema to the relational schema for process logs (RXES [54]) is required. In addition to this, we also developed a software tool that can bridge from RXES to the standard format for process logs (XES [45]). We plan to use this tool with the approach from [55] in order to make it fully compatible with the RXES standard. This will allow to use the SQL language to not only make simple queries on top of the Camunda logs, but also to perform process mining on based on these logs.

Process adapter. The process adapter module (cf. Figure 2) that we describe in the framework is yet to be implemented. This module is developed as an intermediate component between the process monitor and the Camunda engine. It is able to correct slight deviations in the running process, without stopping the workflow. We allow two correction operations for correcting possible deviations: (1) Schedule shifting, and (2) Rescheduling. *Schedule shifting* is the basic operation for correcting delayed activity executions. It simply shifts the starting time of the delayed activity and the subsequent activities if no resource conflict occurs for the resources allocated for these activities. Otherwise, the process adapter communicates to the reasoner for a *rescheduling*. Note that, a rescheduling operation aims at minimal change in schedule, i.e., the starting times and the resource allocations of the affected activities will be altered as least as possible.

Connection to ontology. The current prototype uses Camunda databases as a source of data. We are going to connect our prototype to our engineering domain ontology [1] which is continuously improved.

User interface. A component for visualizing schedules is mandatory in our prototype. Currently, we show schedules as a Gantt chart. Users are going to be able to enter their scheduling preferences on this chart(e.g., constraining the start time of an activity, constraining resource allocation to one particular resource set, locking separate activity schedules for rescheduling purposes, etc.) in an interactive way. Moreover, we support mining and monitoring techniques whose results are generated out of unstructured data. Presenting this data in the most readable way is essential for the user to utilize it in management and other decision-making.

The Camunda BPMS along with our custom extensions has been deployed on a server and can be used by following this link: http://camunda.ai.wu.ac.at:8080/camunda. Credentials for project managers – administrative users with higher privileges – are ‘demo’ and ‘demo’, respectively user name and password; standard users can access with username same as their name and the string ’password’ as password. A screencast of the prototype can be found in http://camunda.ai.wu.ac.at/shapeworks/video.html.

6 Related Work

The existing work on similar frameworks are from safety management [56–58], and decision support domains [59, 60]. To best of our knowledge, there is no framework addressing all the seven requirements (cf. Section 3) that we identify. Therefore, we elaborate on the supporting literature.

Bowen and Stavridou [57] detail the standards concerned with the development of safety-critical systems, and the software in such systems. They identify the challenges of safety-critical computer systems, define the measures for the correctness of such systems and its relevance to several industrial application areas of, e.g. formal methods in railway systems, which is crucial for rigorous and coherent process management.

De Medeiros et al. [60] investigate the core building blocks necessary to enable semantic process mining techniques/tools. They conclude that semantic process mining techniques improve the conventional ones by using three elements, i.e., ontologies, model references from elements in logs/models to concepts in ontologies, and reasoners. Our framework supports such a high-level semantic analysis through our integrated semantic model and the reasoner module.

Wilke et al. [56] describe a framework for a holistic risk assessment in airport operations. They focus on coordination and cooperation of various actors through a process model derived in BPM, which helps determination of causal factors underlying operation failures, and detection and evaluation of unexpected changes. The holistic consideration of operations handling rules and regulations of their particular domain serves for ensuring compliance. Daramola et al. [58] describe the use of ontologies in a scenario requiring identification of security threats and recommendation of defence actions. Their approach not only help the quick discovery of hidden security threats but also recommend appropriate countermeasures via their semantic framework. By following this approach, they minimize the human effort and enable the formulation of requirements in a consistent way. In our framework we similarly monitor our ontology for compliance checking by querying the ontology via the queries derived from regulations.

Van der Aalst [61] introduced a Petri net based scheduling approach to show that the Petri net formalism can be used to model activities, resources and temporal constraints with non-cyclic processes. Several attempts have also been done to implement the problem as a constraint satisfaction problem. For instance, Senkul and Toroslu [62] developed an architecture to specify resource allocation constraints and a Constraint Programming (CP) approach to schedule a workflow according to the constraints defined for the tasks. Our framework addresses resource allocation via the reasoner module using ASP.

Zahoransky et al. [59] investigate operational resilience of process management. Their approach is proposed as a complementary approach to risk-aware BPM systems, which focuses on detecting the resilience properties of processes based on measures by mining process-logs for decision support to increase process resilience, and therefore provide flexibility. This approach enables agility in run-time and provides a solid foundation for process execution reliability. We address these aspects in our integrated system via the process miner and the process adapter.

7 Conclusions

In this deliverable we have explored the challenges of safety-critical human- and data-centric process management in engineering projects which are subject to a large amount of regulations and restrictions, i.e. temporal, resource-related and logistical restrictions, as described in the industry scenario (cf. Section 3). Moreover, we have detailed the implementation of our prototype (cf. Section 4 and 5). A literature survey on similar frameworks is also provided (cf. Section 6).

We have identified some open questions that can be investigated further concerning the WP2:

Declarative process exception handling mechanisms Russell et al. [63] identify three strategies in regard to process exception handling: *No action*, *rollback*, and *compensation*: A rollback is an operation which returns the process execution to some previous state, and a compensation is the action taken to recover from error or cope with a change of plan. In long running business process instances, rollback operation requires a compensation process that undos the effects of executed actions up to the rollback state, because parts of a transaction (e.g., communications with external agents) are inherently impossible to undo. In practice, these processes not only are hardcoded for each rollback scenario but also require regular maintenance when the main process is updated. Storing transactions while process execution in a machine readable manner and defining a formal mechanism to generate compensation processes for supporting exception handling mechanisms when necessary via automated planning would be very useful in case of failure scenarios.

Hierarchical allocation of resources We have generalized our resource allocation technique for tackling the needs of safety critical engineering projects in BPM [53]. However, extending this generalized approach towards a hierarchical method would be beneficial for addressing large resource allocation problems, e.g., in three layers: (1) Allocation in processes, (2) allocation in projects, and (3) allocation in organizations. Note that an organization executes multiple projects and each project consists of multiple processes.

References

1 Jan Mendling Axel Polleres Simon Steyskal, Cristina Cabanillas. Engineering Domain Ontology. Project deliverable (d4.1-d4.4), Siemens, 2016.

2 Jan Mendling Axel Polleres Giray Havur, Cristina Cabanillas. State-of-the art report on existing models for processes, resources, constraints and security and their underlying formalisms. Project deliverable (d2.1), Vienna University of Economics and Business, 2015.

3 Jan Mendling Axel Polleres Giray Havur, Cristina Cabanillas. Unified model for semantic models for mining and monitoring process-relevant data. Project deliverable (d2.2), Vienna University of Economics and Business, 2015.

4 Jan Mendling Axel Polleres Saimir Bala, Cristina Cabanillas. Requirements for process, resource and compliance rules extraction from text. Project deliverable (d3.1), Vienna University of Economics and Business, 2015.

5 Jan Mendling Axel Polleres Saimir Bala, Cristina Cabanillas. Mining processes, resource consumption and witnesses for task completion from logs. Project deliverable (d3.2), Vienna University of Economics and Business, 2015.

6 Jan Mendling Axel Polleres Saimir Bala, Cristina Cabanillas. Combined method for mining and extracting processes, related events and compliance rules from unstructured data. Project deliverable (d3.3), Vienna University of Economics and Business, 2016.

7 Alois Haselböck Simon Sperl. Rules catalogue for necessary process adaptations. Project deliverable (d5.1), Siemens, 2016.

8 Simon Sperl. Integration of re-configuration/re-scheduling algorithms into process architecture. Project deliverable (d5.2), Siemens, 2016.

9 Jan Mendling Axel Polleres Giray Havur, Cristina Cabanillas. Resource and data management service architecture. SHAPE project deliverable.

10 Gerhard Fleischanderl, Gerhard E. Friedrich, Alois Haselböck, Herwig Schreiner, and Markus Stumptner. Configuring Large Systems Using Generative Constraint Satisfaction. *IEEE Intelligent Systems*, 13(4):59–68, 1998.

11 OMG. BPMN 2.0. Recommendation, OMG, 2011.

12 Guido Governatori and Shazia Sadiq. The Journey to Business Process Compliance. In *Handbook of Research on BPM*, pages 426–454. IGI Global, 2009.

13 Cristina Cabanillas, Manuel Resinas, Adela del Río-Ortega, and Antonio Ruiz-Cortés. Specification and Automated Design-Time Analysis of the Business Process Human Resource Perspective. *Inf. Syst.*, 52:55–82, 2015.

14 M. Bozzano and A. Villafiorita. *Design and safety assessment of critical systems.* CRC Press Taylor & Francis Group, 2010.

15 Wil van der Aalst. *Process mining: discovery, conformance and enhancement of business processes.* Springer-Verlag Berlin Heidelberg, 2011.

16 Mariano F. Lopez, Asuncion G. Perez, and Natalia Juristo. METHONTOLOGY: from Ontological Art towards Ontological Engineering. In *AAAI97 Symposium*, pages 33–40, 1997.

17 Cristina Cabanillas, Alois Haselböck, Jan Mendling, Axel Polleres, Simon Sperl, and Simon Steyskal. Engineering Domain Ontology. SHAPE project deliverable.

18 Simon Steyskal and Axel Polleres. Defining expressive access policies for linked data using the ODRL ontology 2.0. In *SEMANTICS 2014*, pages 20–23, 2014.

19 Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Boris Villazón-Terrazas. How to write and use the Ontology Requirements Specification Document. In *On the move to meaningful internet systems: OTM 2009*, pages 966–982. Springer, 2009.

20 Nick Russell, Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and David Edmond. Workflow Resource Patterns: Identification, Representation and Tool Support. In *CAiSE*, pages 216–232, 2005.

21 Cristina Cabanillas, Manuel Resinas, Jan Mendling, and Antonio Ruiz Cortés. Automated team selection and compliance checking in business processes. In *Proceedings of the 2015 International Conference on Software and System Process, ICSSP 2015, Tallinn, Estonia, August 24 - 26, 2015*, pages 42–51, 2015.

22 Cristina Cabanillas, Manuel Resinas, and Antonio Ruiz-Cortés. RAL: A High-Level User-Oriented Resource Assignment Language for Business Processes. In *Business Process Management Workshops (BPD'11)*, pages 50–61, 2011.

23 Cristina Cabanillas, David Knuplesch, Manuel Resinas, Manfred Reichert, Jan Mendling, and Antonio Ruiz-Cortés. RALph: A Graphical Notation for Resource Assignments in Business Processes. In *CAiSE*, volume 9097, pages 53–68. Springer, 2015.

24 WM Zuberek. Timed petri nets definitions, properties, and applications. *Microelectronics Reliability*, 31(4):627–644, 1991.

25 Giray Havur, Cristina Cabanillas, Jan Mendling, and Axel Polleres. Automated Resource Allocation in Business Processes with Answer Set Programming. In *BPM Workshops (BPI)*, page In press, 2015.

26 Remco M. Dijkman, Marlon Dumas, and Chun Ouyang. Semantics and analysis of business process models in BPMN. *Information & Software Technology*, 50(12):1281–1294, 2008.

27 Remco M Dijkman, Marlon Dumas, and Chun Ouyang. Formal semantics and analysis of BPMN process models using Petri nets. Technical Report 7115, Queensland University of Technology, 2007.

28 Wil MP Van der Aalst. The application of petri nets to workflow management. *Journal of circuits, systems, and computers*, 8(01):21–66, 1998.

29 Niels Lohmann, Eric Verbeek, and Remco Dijkman. Petri Net Transformations for Business Processes - A Survey. *Transactions on Petri Nets and Other Models of Concurrency II*, 2:46–63, 2009.

30 Michael Weber and Ekkart Kindler. The petri net markup language. In Hartmut Ehrig, Wolfgang Reisig, Grzegorz Rozenberg, and Herbert Weber, editors, *Petri Net Technology for Communication-Based Systems*, volume 2472 of *Lecture Notes in Computer Science*, pages 124–144. Springer, 2003.

31 Linh Thao Ly, Fabrizio M. Maggi, Marco Montali, Stefanie Rinderle-Ma, and W.M.P. van der Aalst. Compliance monitoring in business processes: Functionalities, application, and tool-support. 54:209–234, March 2015.

32 Julia Fuchsbauer. How to manage Processes according to the European Norm 50126 (EN 50126). Bachelor thesis, 2015.

33 Holger Knublauch and Arthur Ryman. Shapes Constraint Language (SHACL). Working Draft (work in progress), W3C, 2016. https://www.w3.org/TR/shacl/.

34 Khalil Riad Bouzidi, Catherine Faron-Zucker, Bruno Fies, and Nhan Le Thanh. An ontological approach for modeling technical standards for compliance checking. In *Web Reasoning and Rule Systems*, pages 244–249. Springer, 2011.

35 Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. *Answer Set Solving in Practice.* Morgan & Claypool Publishers, 2012.

36 Gerhard Brewka, Thomas Eiter, and Mirosław Truszczyński. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.

37 Agostino Dovier, Andrea Formisano, and Enrico Pontelli. A comparison of clp (fd) and asp solutions to np-complete problems. In *Logic Programming*, pages 67–82. Springer, 2005.

38 Markus Aschinger, Conrad Drescher, Gerhard Friedrich, Georg Gottlob, Peter Jeavons, Anna Ryabokon, and Evgenij Thorstensen. Optimization methods for the partner units problem. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 4–19. Springer, 2011.

39 Francesco Calimeri, Martin Gebser, Marco Maratea, and Francesco Ricca. Design and results of the fifth answer set programming competition. *Artificial Intelligence*, 231, 2016.

40 Thomas Eiter, Giovambattista Ianni, Thomas Krennwallner, and Axel Polleres. Rules and Ontologies for the Semantic Web. In *Reasoning Web 2008*, volume 5224, pages 1–53. San Servolo Island, Venice, Italy, 2008.

41 Wil M. P. van der Aalst and Arthur H. M. ter Hofstede. YAWL: Yet Another Workflow Language. *Inf. Syst.*, 30(4):245–275, 2005.

42 L. J. R. Stroppi, O. Chiotti, and P. D. Villarreal. A BPMN 2.0 Extension to Define the Resource Perspective of Business Process Models. In *CIbS'11*, 2011.

43 Cristina Cabanillas, Manuel Resinas, Jan Mendling, and Antonio Ruiz Cortés. Automated team selection and compliance checking in business processes. In *ICSSP*, pages 42–51, 2015.

44 B. F. Van Dongen, A. K A De Medeiros, H. M W Verbeek, A. J M M Weijters, and W. M P Van Der Aalst. The ProM framework: A new era in process mining tool support. In *Lect. Notes Comput. Sci.*, volume 3536, pages 444–454. Springer, 2005.

45 H. M W Verbeek, Joos C A M Buijs, Boudewijn F. Van Dongen, and Wil M P Van Der Aalst. XES, XESame, and ProM 6. In *Lect. Notes Bus. Inf. Process.*, volume 72 LNBIP, pages 60–75. Springer, 2011.

46 Ekkart Kindler, Vladimir Rubin, and Wilhelm Schäfer. Activity Mining for Discovering Software Process Models. *Softw. Eng.*, 79:175–180, 2006.

47 Wouter Poncin, Alexander Serebrenik, and Mark Van Den Brand. Process Mining Software Repositories. *2011 15th Eur. Conf. Softw. Maint. Reengineering*, pages 5–14, 2011.

48 Claudio Di Ciccio, Massimo Mecella, Monica Scannapieco, Diego Zardetto, and Tiziana Catarci. MailOfMine - Analyzing mail messages for mining artful collaborative processes. *Lect. Notes Bus. Inf. Process.*, 116 LNBIP:55–81, 2012.

49 Saimir Bala, Cristina Cabanillas, Jan Mendling, Andreas Rogge-Solti, and Axel Polleres. Mining Project-Oriented Business Processes. In *BPM*, pages 425–440, 2015.

50 C Michael Pilato, Ben Collins-Sussman, and Brian W Fitzpatrick. *Version control with subversion.* " O'Reilly Media, Inc.", 2008.

51 Linus Torvalds and Junio Hamano. Git: Fast version control system. *URL http//git-scm.com*, 2010.

52 Cristina Cabanillas, Saimir Bala, Jan Mendling, and Axel Polleres. Combined method for mining and extracting processes, related events and compliance rules from unstructured data. Technical report, WU Vienna, 2016.

53 Giray Havur, Cristina Cabanillas, Jan Mendling, and Axel Polleres. Resource Allocation with Dependencies in Business Processes Management Systems. In *BPM Forum*, page In press, 2016.

54 B F van Dongen and Sh Shabani. Relational XES : Data Management for Process Mining. *BPM Cent. Rep. BPM-15-02*, 2015.

55 Stefan Schönig, Cristina Cabanillas, Stefan Jablonski, and Jan Mendling. Mining the Organisational Perspective in Agile Business Processes. In *BPMDS*, pages 37–52, 2015.

56 Sabine Wilke, Arnab Majumdar, and Washington Y Ochieng. Airport surface operations: A holistic framework for operations modeling and risk management. *Safety Science*, 63:18–33, 2014.

57 Jonathan Bowen and Victoria Stavridou. Safety-critical systems, formal methods and standards. *Software Engineering Journal*, 8(4):189–209, 1993.

58 Olawande Daramola, Guttorm Sindre, and Thomas Moser. A tool-based semantic framework for security requirements specification. *J. UCS*, 19(13):1940–1962, 2013.

59 Richard M Zahoransky, Christian Brenig, and Thomas Koslowski. Towards a process-centered resilience framework. In *Availability, Reliability and Security (ARES), 2015 10th International Conference on*, pages 266–273. IEEE, 2015.

60 Ana Karla Alves de Medeiros, Wil Van der Aalst, and Carlos Pedrinaci. Semantic process mining tools: core building blocks. 2008.

61 W.M.P. van der Aalst. Petri net based scheduling. *Operations-Research-Spektrum*, 18(4):219–229, 1996.

62 Pinar Senkul and Ismail H. Toroslu. An Architecture for Workflow Scheduling Under Resource Allocation Constraints. *Inf. Syst.*, 30(5):399–422, July 2005.

63 Nick Russell, Wil van der Aalst, and Arthur ter Hofstede. Workflow exception patterns. In *International Conference on Advanced Information Systems Engineering*, pages 288–302. Springer, 2006.