Complex Systems Informatics and Modeling Quarterly
CSIMQ, Issue 8, September/October 2016, Pages 15–27
Published online by RTU Press, https://csimq-journals.rtu.lv
https://doi.org/10.7250/csimq.2016-8.02
ISSN: 2255-9922 online

Exploring the Potential of Dynamic Perspective Taking on Business Processes

Florian Krenn and Christian Stary

Department of Business Information Systems-Communications Engineering,
Johannes Kepler University Linz, Altenberger Straße 69, 4040 Linz, Austria

{Florian.Krenn, Christian.Stary}@jku.at

Abstract. Although many organizations have started to work with business process models in their operational practice, they have not explored the entire potential of intertwining business process modeling with organizational development. Process specifications contain workflows that require execution, in order to achieve business objectives and support business operation effectively. With the advent of Subject-oriented and Social Business Process Management, communication and stakeholder interaction have become novel perspectives on how to design and implement processes. They go beyond formal responsibilities encoded in functional roles, and are not very common across organizational hierarchies. However, stakeholders, including organizational developers and IT specialists, can be supported looking at processes and their execution from either perspective, namely, from a traditional one, focusing on functions and task accomplishment, and from an interactional perspective, focusing on communication among stakeholders and system interactions. The introduced dual-mode workflow execution engine UeberFlow allows considering both perspectives during process runtime, thus, checking operational completeness from either perspective. Stakeholders can start modeling with a perspective they are familiar with and subsequently proceed with the another one by switching dynamically to an alternate mode of execution. The presented meta-model and architecture of such a dual mode support tool enables coupling business process management directly with organizational development.

Keywords: Polymorph execution, SBPM, S-BPM, function flow, interaction flow.

1 Introduction

Meanwhile many organizations consider Business Process Management (BPM) [1] a valuable approach to sustain on volatile markets. However, they still struggle to understand its capacity, and, thus, to fully implement the concept, in particular, when intertwining modeling and execution could support the dynamics of business development [2]. These findings triggered research and technology developments for interactive stakeholder support to facilitate capturing of information relevant for modeling and generating executable models [3], [4]. Of utmost importance seems to be peer-to-peer relationships in today's organizations [5]. Starting with management, another mental shift seems to be required, similar to Frederick Taylor's move to the factory from the craft level (ibid.). This time net-worked stakeholders rather than functional role carriers move to the center of interest.

Traditionally, peer-to-peer relationships have only been of interest in case the efficiency of producing goods or services has been affected (cf. [6] analyzing [7]). Hence, traditional models of organizing work capture technical or functional role relationships [2]. Suchman [4] in her study has found out, ‘that although system designers recognize the centrality of procedural tasks in the office, they tend to ignore the actual work involved when accomplishing those tasks’ (p. 320). She found this approach in line with Taylor’s separation of planning and execution, the latter representing the functional steps or processes required to accomplish tasks.

Such a functional perspective on processes has been questioned (cf. [8]) as it primarily hinders developing organizations when they operate under volatile conditions or in innovative sectors [9]. Moreover, knowledge about work and, thus, change is not embodied in the workers’ mind [10]. Consequently, informed organizational development can only be triggered by certain stakeholders (i.e. management) which makes forming holistic organizations unlikely [11] – structural coupling between organizational communication (social system) and perception (psychic system) is hindered [12]. This coupling seems to be essential, as reorganizing work through stakeholders requires knowledge on interaction and tasks [12]. In addition, stakeholders need qualification and tools to express both types of knowledge for actively participating in Business Process Management projects [14], [15]. Figure 1 shows a simple process from a functional (task-oriented) and interactional perspective, in order to illustrate the different points of view. The left side of the figure shows the process from a subject-oriented perspective. The three process participants (Requester, Head of Department, and Cost Center) are depicted including their interaction and information exchange. The other perspective shows the same process from a functional point of view, focusing on the sequence of tasks in this business process.

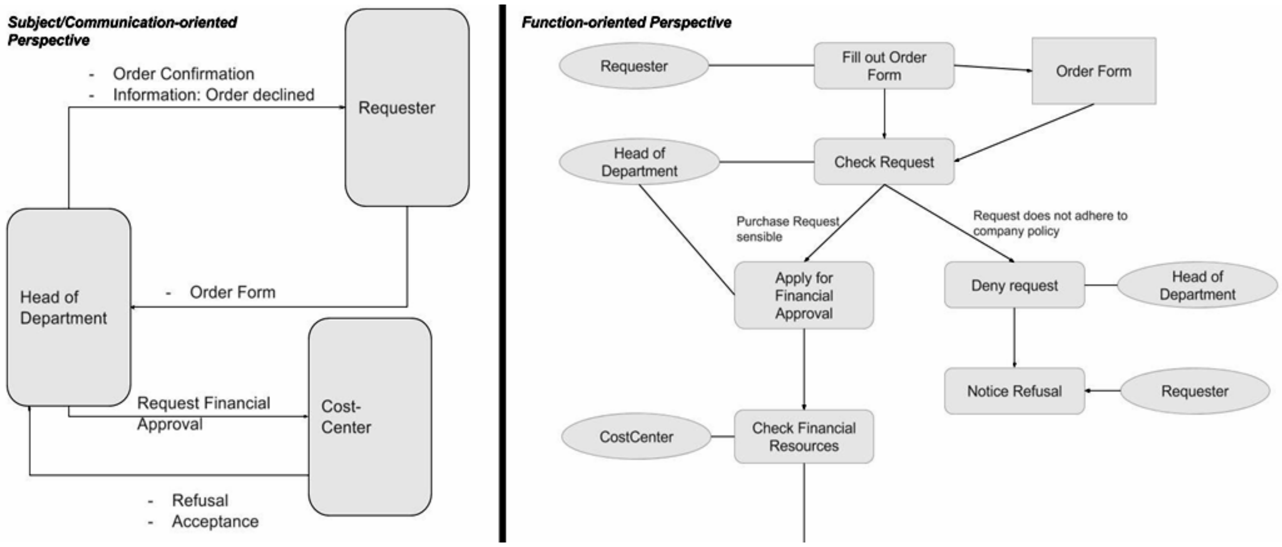

Subject/Communication-oriented Perspective
- Order Confirmation
- Information: Order declined
Requester
Head of Department
- Order Form
Request Financial Approval
Cost-Center
- Refusal
- Acceptance
Function-oriented Perspective
Requester
Fill out Order Form
Order Form
Head of Department
Check Request
Purchase Request sensible
Request does not adhere to company policy
Apply for Financial Approval
Deny request
Head of Department
Notice Refusal
Requester
CostCenter
Check Financial Resources

Figure 1. One process – two perspectives: function and interaction flow

This work first aims to provide an understanding of barriers still hindering organization-relevant stakeholders to utilize BPM system capabilities. Secondly, in line with the findings reported above, we introduce two relevant and complimentary perspectives when specifying and, finally, executing process specifications, namely, function-based and interaction-based workflows. Function-based workflow specifications allow focusing on steps required for task accomplishment, whereas interaction-based execution puts the exchange of messages or business objects between self-contained behavior entities (role actors or applications) to the center of interest.

The development of such a dual approach to execution has been triggered by Social BPM developments (SBPM) which take into account the social nature of executing and re-designing work processes [5]. Besides embedding Social Media into the BPM lifecycle, such as enriching process models and tagging process elements [16], [17], or annotating social interactions [18], the required interaction among stakeholders for achieving business objectives can be encoded directly into process models [3], [15]. Such an endeavor provides the opportunity to view functional behavior from the social perspective in an integrated way, as stakeholders interact when accomplishing their work tasks and, thereby complete processes. A corresponding workflow engine enables stakeholders (project managers, developers, users) to experience processes live and reflect on interaction and functional behavior when implementing them in an organization.

We have structured the paper as follows: Section 2 discusses various perspectives on processes and process specifications. The results allow arguing for capturing and following various angles of looking at process specifications, and, finally, modes of execution. Section 3 details the conceptual meta-model when representing a functional and an interactional viewpoint on process specifications, and the architecture of a system preserving the interactional and functional flow of task accomplishment when executing these specifications. Section 4 demonstrates how such a tool can be implemented and used in BPM practice. Section 5 closes the paper, wrapping up research objectives and results, and revealing topics of further research.

2 Perspectives on Process Specifications

In this section, we review selected BPM approaches that refer to or encode perspectives on business processes. It allows arguing for developing or preserving perspectives on business processes as they facilitate understanding behavior patterns of organizations when tasks are accomplished or strategic objectives need to be met.

User roles play a crucial role on how to look at process specifications, as they do not only encapsulate targeted and self-contained behavior sequences, but also refer to qualification profiles of human actors or to requirements applications need to meet. Recently, Trkman et al. identified roles as essential context of activities [19]. The acquisition and representation method the authors propose should increase the understanding of the execution order of process steps and, thus, integration dependencies for effective execution support. Thereby, user stories are put into relation to business process model activity elements. A user story is a brief statement involving stakeholder roles and activities in the form of I as a <user role> perform <function>. In principle each functional activity of a process can be contextualized by such a statement. However, a set of such statements does do not lead to complete or cohesive functional sequences. In the study, the standard notation for modeling business processes, the Business Process Model and Notation (BPMN) (www.bpmn.org) has been used to represent processes. The authors considered the activity element as ‘the most important element since it is associated with a user story’ [19]. An activity can be either an atomic or non-atomic (compound) unit of work that an organization performs in its business processes. It provides insights into the “What” an organization is doing. This prominent denotation of an activity indicates the primary perspective on processes, namely, a function-oriented one.

However, ‘its surrounding elements such as swim lanes, XOR gateways, and flow arrows’ were considered ‘important for understanding the dependencies’ (ibid.). Thereby, swim lanes or pools serve as a graphical container for partitioning a set of activities from other activities. A pool represents a participant in a collaboration (e.g., a department), whereas a lane is a partition that is used to organize and categorize activities within the pool (e.g., a specific organizational role within a department). In this way, swim lanes provide insight into “Who” performs a specific activity and, thus, a constitutive element of taking an interactional perspective.

A BPMN business process model is a set of activities that represent the steps required to achieve a business objective. Its name should refer to the “Why” of the represented business operation. The use of these BPMN elements facilitates the integration of swim lane (Who) and activity (What), according to the coupling of the user story items ‘user role’ and ‘function’.

Hence, the functional and role perspectives are complimentary when developing a contextual model of business processes.

Since the researchers aimed at intelligibility of models for stakeholders, a level of abstraction was selected that allowed for the analysis of the organization's business processes rather than for executing the models automatically. Nevertheless, this case shows how developers take a function-centered perspective of model representations when working with stakeholders. This finding is in line with the results presented in [20]. When working with students, the researchers could identify the initial approach novices are likely to take, namely, to represent processes are flowcharts. 72% of the novices conveyed process information in a diagram detailing the steps as boxes, and giving their order by connecting them with arrows. These empirical findings allow concluding that activity-centered descriptions, as already provided by ARIS [21] are constitutive elements of business process specifications.

Mattos et al. investigated factors influencing activities in each instance of a business process [13]. They referred to several context elements of business operations: environment, people, technologies organization of work, and external factors. The authors were interested in difficulties caused by these factors when business processes are executed. They intended to identify and group contextual elements, enabling the description of a situation of an activity performed in a business process. Such a specification could support the adaptation of a running process, in order to achieve better results for the organization.

Rather than giving a formal model for that, Pinggera et al. refer to the situation when specifications come into being [22]. Enhancing the understanding of how process models are created could help identifying different modeling styles, and, thus, different perspectives on process specifications. The data of 115 students of courses on business process management working on two modeling tasks revealed three distinct modeling styles that occurred independently of the concrete modeling task. This exploratory study allowed researchers categorizing “efficient modeling”, i.e., quickly of adding elements to the model, “layout-driven modeling”, i.e., spending much time in creating a comprehensible layout while being less efficient in creating the model, and “intermediate modeling” that is neither particularly efficient nor invests particularly into model layout. In addition, they found modelers following a “layout-driven modeling style” were investing more work into correcting a model than modelers following other styles.

As the study revealed modelers sticking to the same modeling style in all tasks and modelers following different modeling styles in different tasks, we can conclude that a particular modeling style depends on both modeler- and task-specific characteristics. From data referring to the time needed to think about the modeling task and the rate of adding elements to a model, we can further conclude that modeler preferences or capabilities have stronger impact on following a certain modeling style than the modeling task itself. This also holds for the effort spent on the layout of a model. Interestingly, this effort seems to be independent of the perceived complexity of the modeling task.

According to the findings of [22], the perception of a task difficulty by a modeler seems to correlate with the probability of a modeler expecting difficulties in the course of modeling, and the necessity to rework a model. However, the time spent for creating a model seems to be largely independent of the perceived complexity of the task. Overall, the complexity of a modeling task and the way a modeler perceives it seem to be essential influence factors. For both parameters, focusing on a certain modeling perspective could influence the effectiveness or efficiency of modeling. Hence, the complexity of a modeling task could be reduced for a modeler by switching from a fully loaded functional flow model to a communication-oriented flow of control, simplifying the overall behavior specification. Hereby, the modeler could find a particular viewpoint on modeling more convenient than others, and, thus, more effective to handle a certain case.

Gromoff et al. targeted real-time business process generation, as required for increasing the agility of organizations [8]. They simulated the process of generating models by starting with particular business requests and utilizing the automated execution capability of subject-oriented process specification. The simulation resulted in specific architectures that were either approved or rejected/corrected by stakeholders being in charge for this development. The generation process triggered requirement specification of supporting resources referring to the organizational implementation of a process, such as intellectual and professional skills, and to its technical implementation [3]. Once these requirements can be met through proper services or service provision, a model can be executed and made available to stakeholders being responsible for or involved in the specified business process.

Duarte et al. also investigated the transformation of operational (running) business processes into software execution support from a methodological perspective [9]. Although the methodology promotes the usage of reference models, its four distinct phases and abstraction levels can also be applied for creating and operating process specifications. Recognizing in a special phase the diversity of specifying business processes of an organization the methodology proposes to consider the specific characteristics of an organization at hand, and to tailor the process specifications accordingly. Then, technical systems are selected to implement the process for operation support.

The diversification of process specifications is already supported by several concepts, as they correspond to perspectives on specifications [9]. A core concept is called Instantaneously Available Organization. It promotes templates for detailing process specifications. Enriching traditional reference modeling approaches it also provides role descriptors, which shed a different light on specifications. Another concept also corresponds to a perspective on processes. It is termed Organizational Aspect and concerns orthogonal characteristics an organization can exhibit, e.g., making decisions transparent to all concerned stakeholders. In this way, strategic objectives and cultural issues can be represented and assured through specifications.

The Process Framework [9] refers to the various business implementation methodology (BIM) life-cycle stages and, thus, a series of state-specific specifications. This set of process models and tools enables the development project stakeholders to manage business processes at different states, including activities that can be performed on included process (reference) models, a set of actors that can perform activities, and finally, a business ontology.

The concept of Orchestrated Business Object concerns the execution, as it refers to the software implementation of a business entity and its associated functionalities including operational data. These pieces of software implement business entities inside some specific business ontology, as they expose functionality and data to the software implementing these concepts. In accordance to subject orientation [3] each Orchestrated Business Object (subject/service) is a black-box, and should be interchangeable with other services implementing the same business entity. This concept enables to implement a process utilizing different software applications that are orchestrated in terms of services as specified in the process model. Such an abstraction decouples modeling the organization of work from technical implementation capacities and complement functional specifications with the interactional perspective for process execution. Not only the resources for implementing processes become interchangeable, but also the exchange process itself does not cause any disruption in the organizational behavior. Hence, coupling the functional perspective on business processes with an interactional one is of two-fold benefit: It reduces the complexity of the modeling process and allows high organizational agility when executing the specifications.

3 Meta Modeling a Dual Perspective Engine

In this section, we introduce a meta-model called “UeberFlow Lang” which facilitates the capturing of the functional and interactional perspective on executable process specifications. We sketch the underlying theory in Section 3.1, before introducing the meta-model in Section 3.2, and its visualization for dual use in Section 3.3.

3.1 Underlying Concepts

UeberFlow Lang builds on the Actor Model of Concurrent Computation as introduced by Carl Hewitt, Peter Bishop, and Richard Steiger in 1973 [23]. The Actor Model is a mathematical model of concurrent computation that treats actors as the primitive of computation. The Actor formalism defines a system's behavior (e.g., control flow and data flow) only in terms of the interaction between actors. An actor is considered as an entity that is able to receive messages and react in a defined manner based on its state. Upon message receipt, an actor can perform the following operations [24]:

- Send a finite number of messages to other actors.
- Create a finite number of new actors.
- Designate the behavior to be used for the next message it receives.
- Based on these mechanisms later implementations of the Actor Model additionally introduced structuring mechanisms of actors in form of hierarchies and supervision strategies.

Although developed in 1973, the Actor Model has significantly gained importance over the past years in the field of software development. The upcoming paradigms like Internet of Things and Cyber Physical Systems have led to a shift in requirements for software applications and workflow implementations. Decentralization of computing, availability, parallelism, elasticity, and flexibility have become important characteristics of modern software applications (cf. http://www.reactivemanifesto.org/). Creating these so-called reactive applications is facilitated by actor-oriented design [24].

3.2 UeberFlow Lang – The Meta Model

Generally, a workflow model comprises of a set of actions or tasks, which are ordered in a certain sequence and performed by workflow participants having certain roles. Workflow specifications in UeberFlow Lang incorporate these workflow elements using three basic building blocks. UeberFlow Lang Workflow specifications, as illustrated in Figure 2, comprise WorkflowUnits, WorkflowSteps, and WorkflowFunctions. Each component defined in the meta-model can be understood as an actor according to the actor model adhering to the defined mechanisms and regulations [24].

WorkflowSpecification. The WorkflowSpecification represents an entirely executable model of the workflow. It acts as container for the WorkflowUnits and stores the relevant meta-data like creation date or access permissions.

WorkflowUnits. WorkflowUnits group process steps according to the responsible role similar to lanes in BPMN or subjects in subject-oriented workflow specifications. For each role participating in the specified workflow a WorkflowUnit is created which contains and supervises all WorkflowSteps the corresponding role is responsible for. Additionally, WorkflowUnit functions as a data space for the underlying WorkflowSteps and WorkflowFunctions. In the course of execution all data accessible by the associated role are made available to the WorkflowSteps via the WorkflowUnit.

WorkflowSteps. WorkflowSteps represent the activities a workflow comprises. The actual execution logic of a WorkflowStep, its prerequisites and results, are solely defined by its WorkflowFunctions. Each WorkflowStep contains a sequence of WorkflowFunctions which are executed sequentially, when the corresponding step is triggered. WorkflowFunctions can be assigned to a WorkflowStep without any limitations concerning order or quantity. The execution of a step is complete, once all its WorkflowFunctions have been executed successfully.

WorkflowFunctions. WorkflowFunctions are the most fine-grained units of execution in the UeberFlow Lang meta-model, and the only truly active component from the workflow's perspective. Each WorkflowFunction represents an atomic action of workflow execution. In order to define the workflow execution logic on a very fine-grained level, for each WorkflowFunction an optional condition can be specified which limits the set of situations the WorkflowFunction is triggered based on current instance data.

Since WorkflowFunctions encapsulates all actions of a workflow specification, different types of WorkflowFunctions are needed for runtime purposes. In the herein presented basic version of UeberFlow Lang six WorkflowFunction-types are defined.

RequireFunction. The RequireFunction allows specifying a set of values required for the execution of subsequent functions defined in the WorkflowStep. These values can either represent an event triggered during the workflow execution or a set of data. The execution of the process step stops until the required values are available for the process unit. The RequireFunction has an optional convert expression, which allows modifying the data before it is made available in the context of the WorkflowStep. Since the RequireFunction completely abstracts from the source of the data it is agnostic to whether the incoming (or already available) data was provided via a message or by the previous step.

ProvideFunction. Upon execution, the ProvideFunction sends a set of values to any WorkflowUnit defined in the workflow. Analogous to the RequireFunction, these provided values can either be a set of data (e.g., completed order form) or an event. Thus, the ProvideFunction can be used in combination with the RequireFunction. It implements asynchronous messaging/data exchange between WorkflowUnits. In this way, data become available for subsequent steps in the same unit.

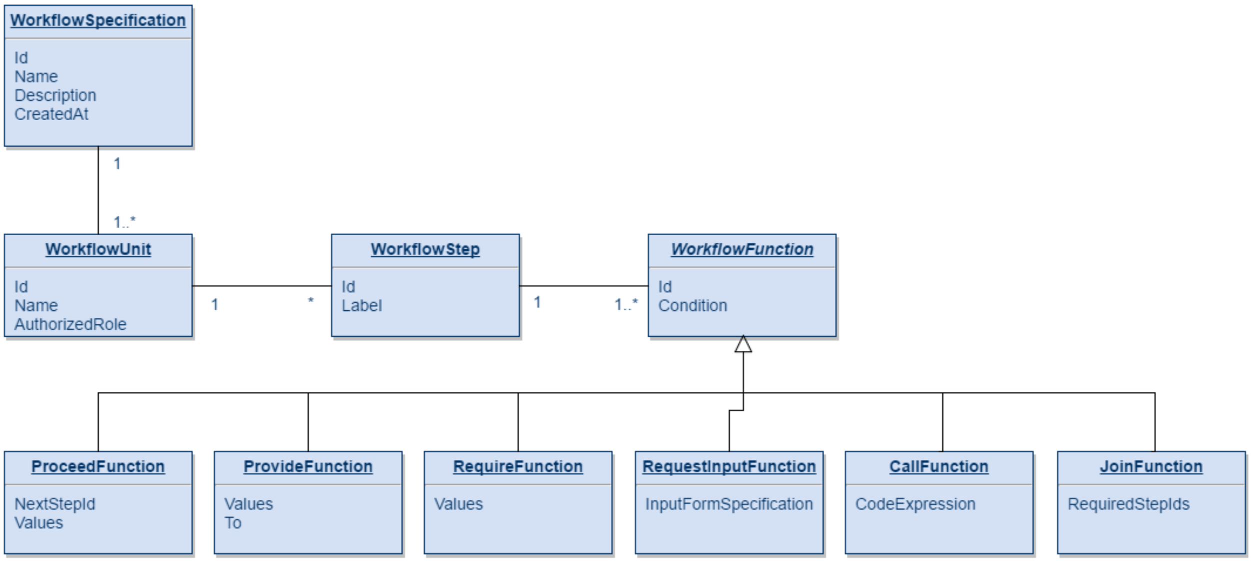


Figure 2. Constructs defined in the UeberFlow meta-model

ProceedFunction. The ProceedFunction triggers the execution of another WorkflowStep. The execution of the current step has not to be complete, in order to trigger the next step, i.e., other functions can be executed after the ProceedFunction has been executed. AND-, OR- and, XOR-Gateways can be implemented by specifying multiple sequential ProceedFunctions and corresponding conditions within one WorkflowStep. Besides simply triggering the execution of a subsequent step, the ProceedFunction offers an alternative to the ProvideFunction. It is also possible to directly pass data to the triggered step. For example, the result of a calculation performed by step A can be passed on the subsequent step B without adding it to the data context of the WorkflowUnit.

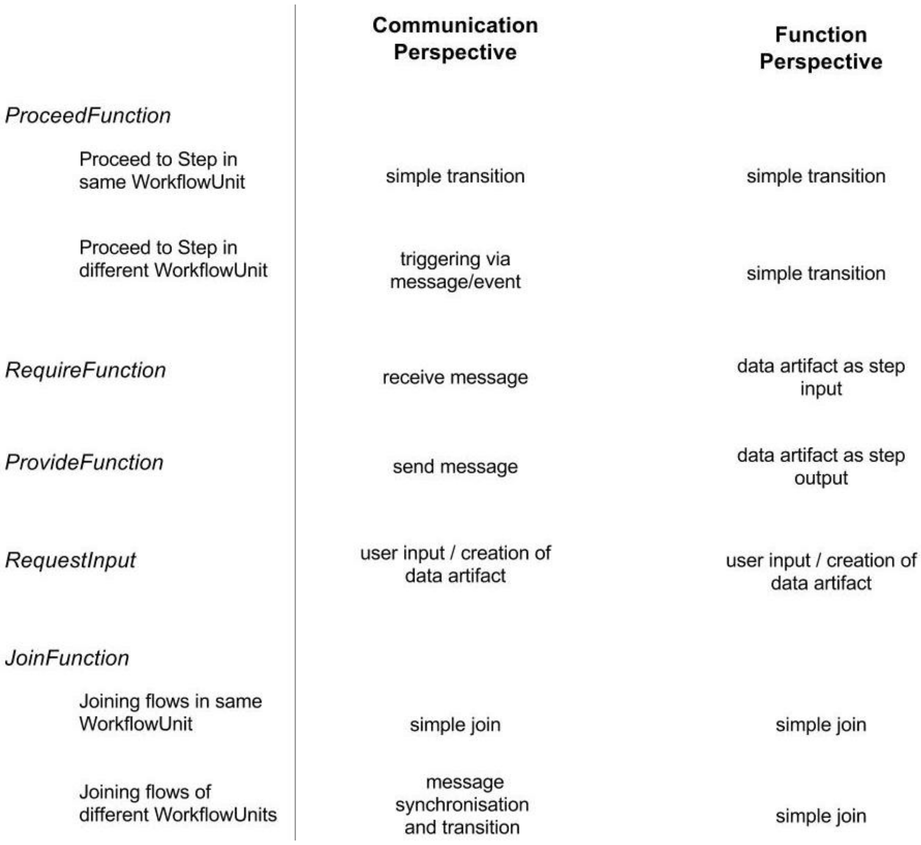
JoinFunction. The JoinFunction enable synchronizing two or more parallel execution flows by halting execution of the containing ProcessStep until all of the defined previous steps have been executed. It is also possible to define a subset of steps required in order to realize a partial join according to the workflow patterns as described by [25]. The JoinFunction does not distinguish between the synchronization of paths within a single WorkflowUnit or synchronizing parallel paths of different WorkflowUnits.

RequestInputFunction. The RequestInputFunction is used to define required user input. Based on a specified input form, the current user (or users) associated with the role of the WorkflowUnit is requested to provide an input. The execution of the WorkflowStep is halted until the required input is provided.

CallFunction. The CallFunction allows extending the workflow capabilities by using external services. Such an extension can be achieved by defining a code snippet, which is interpreted at run-time, once the WorkflowFunction is executed.

3.3 Visualizing the Meta Model for Dual Use

In the previous section, the meta-model of UeberFlow Lang has been introduced without focusing on the visualization aspect of switching perspectives when creating workflow specifications or process models. In order to be able to implement a workflow execution engine and editing support building on UeberFlow Lang, visualization mapping of the meta-model to the function- and communication-oriented perspective has to be defined. In the remainder of the paper, we explain how the UeberFlow Lang concepts can be used to create workflow specification from both perspectives. Figure 3 shows the approach that is used in Section 4 in an exemplary use case.



	Communication Perspective	Function Perspective
ProceedFunction		
Proceed to Step in same WorkflowUnit	simple transition	simple transition
Proceed to Step in different WorkflowUnit	triggering via message/event	simple transition
RequireFunction	receive message	data artifact as step input
ProvideFunction	send message	data artifact as step output
RequestInput	user input / creation of data artifact	user input / creation of data artifact
JoinFunction		
Joining flows in same WorkflowUnit	simple join	simple join
Joining flows of different WorkflowUnits	message synchronisation and transition	simple join

Figure 3. Visualization Mapping

4 Dual-Mode Support

The meta-model introduced in Section 3 has been implemented in a dual-perspective workflow engine and modeling tool called UeberFlow. UeberFlow supports creating, editing, validation, and execution of workflow definitions specified using the UeberFlow Lang meta-model.

Furthermore, the transformation of other workflow model specifications to UeberFlow Lang is supported prototypically. In Section 4.1, an overview of the UeberFlow architecture and implementation design is given. In Section 4.2 we provide a use case scenario exemplifying the application of the developed system.

4.1 System Architecture & Implementation

The UeberFlow system comprises five main modules (see Figure 4):

- The data module implements the designed workflow meta-model and is responsible of persisting workflow specifications and instance data.
- The runtime module is the actual implementation of the engine. It interprets the workflow specifications and executes them accordingly.
- Module transformations (i.e., imports and exports of created workflow specifications) are handled in the Transformation module.
- All execution and transformation functions of UeberFlow are provided using a REST API. The specifications of the respective services are implemented in the WebAPI module.
- The Frontend module contains a web-based implementation of the workflow user interface and a dual-perspective editor for creating UeberFlow specifications.

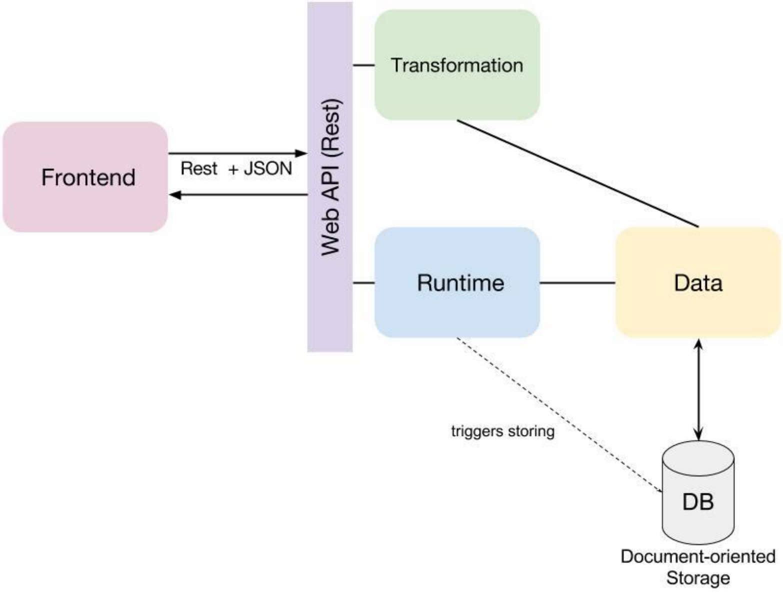


Figure 4. UeberFlow implementation components

In alignment with UeberFlow Lang the implementation of UeberFlow follows the Actor formalism throughout the entire development. For each component of the UeberFlow Lang an Actor has been implemented. For this purpose, the Akka framework (http://akka.io/) has been used as the fundamental approach. Consequently, an instance in UeberFlow is equivalent to all actor instances created in the context of this particular workflow instance. All of those actor instances are aggregated using the actor structuring and supervision mechanisms. By defining a root actor representing the entire instance, it has assigned actor instances. Each actor in UeberFlow is implemented in a way that it can be directly stored in the database comprising its current state. This allows a simple yet efficient persistence and restoring of instances.

4.2 Using UeberFlow

Subsequently, the application of UeberFlow is shown for a scenario based on a business case. The application scenario deals with elicitation, redesign, validation, and execution a notebook ordering process. It involves three process participants: the employee, who requires a new business notebook, the head of the department, who needs to accept or refuse the request, and the IT department, which is in charge of ordering the new device and the first setup before it is handed over to the employee.

A typical process of using UeberFlow includes creating a model using the dual-perspective editor, validating the result using the UeberFlow Validation mode, and adapting the model according to the validation results. When starting from an interactional or communication-oriented perspective one could start defining all roles (i.e., subjects) involved in the process and then model the encapsulated execution flow of a selected subject, focusing on one process participant at a time. This includes the communication with the other process actors and his or her tasks in this role. Figure 5 depicts the UeberFlow Editor showing the communication-oriented perspective of the sample workflow from the employee's viewpoint.

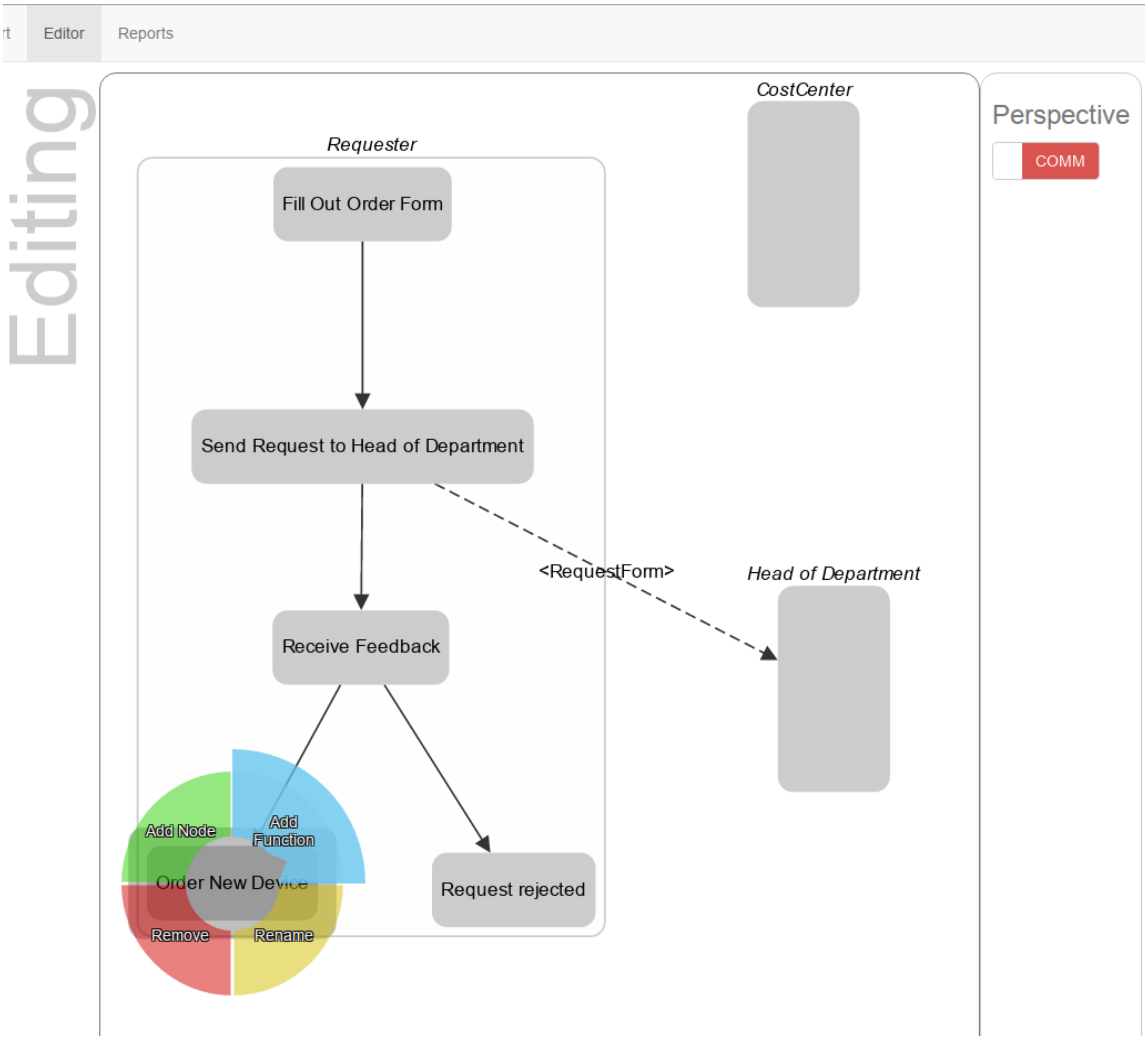


Figure 5. The UeberFlow editor showing the communication-oriented perspective

After having specified the behavior and communication paths of all subjects switching to the functional perspective (see Figure 6) puts the focus on the overall sequence of tasks to be accomplished. This perspective allows a better overview of the entire process compared to focus on the single subject, whilst neglecting the communication aspects.

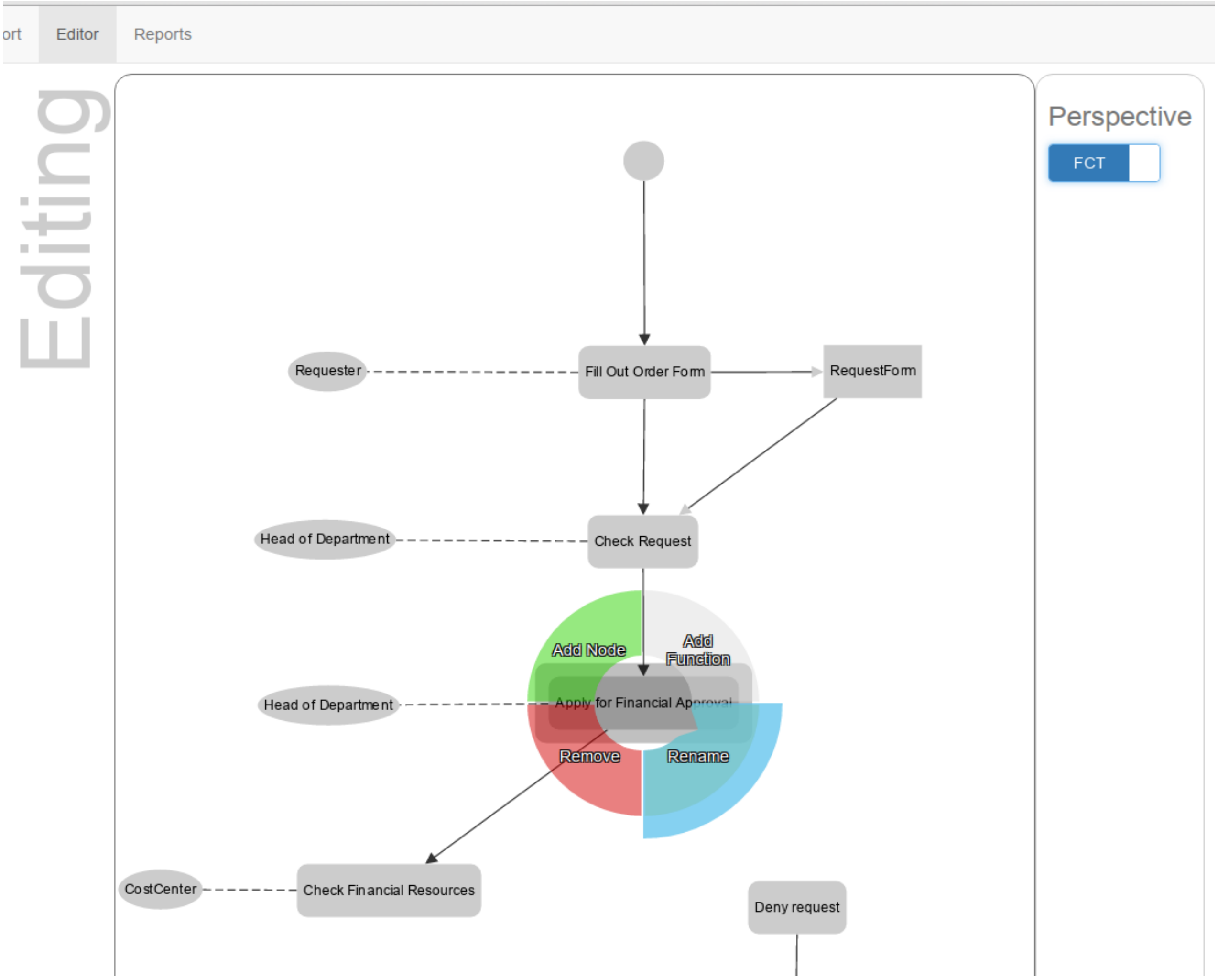
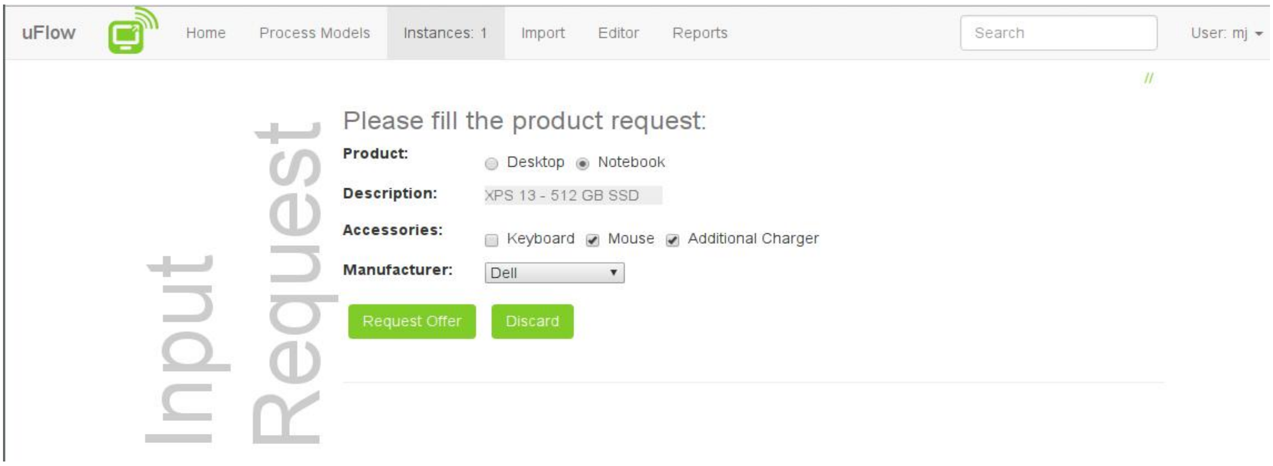


Figure 6. Function-oriented perspective of the sample process

At any time throughout the modeling, the created workflow specification can be executed in the so-called "Evaluation Mode". This execution mode allows a single user stepping through all of the tasks and communication paths, in order to check the semantic correctness of the model (see Figure 7).



Please fill the product request:

Product: ○ Desktop ● Notebook

Description: XPS 13 - 512 GB SSD

Accessories: □ Keyboard ☑ Mouse ☑ Additional Charger

Manufacturer: Dell

Request Offer Discard

Figure 7. Validation and execution in UeberFlow

5 Conclusion and Further Research

Besides function and data flow orientation, the focus on communication and stakeholder interaction has taken hold as major perspective in the design and implementation of processes. Based on these developments, this paper argues for dual-perspective modeling support for workflows, in order to further support the creation of executable process specifications. By designing an actor-based meta-model which can be used as a foundation for providing tool support, a first step towards a dual-perspective workflow specification approach has been made. The provided prototype implementing the designed meta-model and a corresponding editor showed the feasibility of the envisioned approach.

Although the potential of a dual-perspective modeling support is arguable by recent trends and studies in literature, there is yet no study providing empirical insights. Therefore, the next research step will target an empirical evaluation of the herein presented approach.

The extended abstract of this paper is available in [26].

References

[1] M. Weske, “Business Process Management: Concepts, Languages, Architectures,” Springer Publishing Company, Incorporated, 2010.

[2] W.P. Wong, “Business-Process Management: a Proposed Framework for Future Research,” Total Quality Management & Business Excellence, vol. 24, pp. 5–6, 2013. Available: https://doi.org/10.1080/14783363.2013.776773

[3] A. Fleischmann, W. Schmidt, C. Stary, S. Obermeier, and E. Börger, “Subject-Oriented Business Process Management,” Springer Publishing Company, Incorporated, 2012. Available: http://dx.doi.org/10.1007/978-3-642-32392-8

[4] L.A. Suchman, “Office Procedure as Practical Action. Models of Work and System Design,” ACM TA on Office Information Systems, vol. 1, no. 4, pp. 320–328, 1983. Available: http://dx.doi.org/10.1145/357442.357445

[5] J. Barsh, “Innovative Management: a Conversation with Gary Hamel and Lowell Bryan,” in McKinsey Quarterly, no. 1, 24, 2008.

[6] P. Docherty, M. Kira and A.B. Shani, Eds., “Creating Sustainable Work Systems. Developing Social Sustainability,” 2nd ed., Routledge, London, 2009. Available: http://dx.doi.org/10.4324/9780203995389

[7] A.H. ter Hofstede, W. van der Aalst, M. Adams and N. Russell, Eds., “Modern Business Process Automation: YAWL and its support environment,” Springer Science & Business Media, 2009. Available: http://dx.doi.org/10.1108/k.2010.06739fae.002

[8] A. Gromoff, N. Kazantsev, D. Kozhevnikov, M. Ponfilenok and Y. Stavenko, “Newer Approach to Create Flexible Business Architecture of Modern Enterprise,” in Global Journal of Flexible Systems Management, vol. 13, no. 4, pp. 207–215, 2012. Available: http://dx.doi.org/10.1007/s40171-013-0023-z

[9] F.J. Duarte, R.J. Machado and J.M. Fernandes, “BIM: a Methodology to Transform Business Processes into Software Systems,” in Software Quality. Process Automation in Software Development, Springer Berlin Heidelberg, pp. 39–58, 2012. Available: http://dx.doi.org/10.1007/978-3-642-27213-4_4

[10] M. Bahnisch, “Embodied Work, Divided Labour Subjectivity and the Scientific Management of the Body,” in Frederick W. Taylor's 1907 “Lecture on Management,” Body & Society, vol. 6, no. 1, pp. 51–68, 2000. Available: http://dx.doi.org/10.1177/1357034x00006001004

[11] A. Lindbeck and D.J. Snower, “Multitask Learning and the Reorganization of Work: from Tayloristic to Hholistic Organization,” in Journal of labor economics, vol. 18, no. 3, pp. 353–376, 2000. Available: https://doi.org/10.1086/209962

[12] D. Baecker, “The Design of Organizations in Society,” in Niklas Luhmann and Organization Studies, D. Seidl, K.H. Becker, Eds., Liber & Kopenhagen Business School Press, pp. 181–212, 2006.

[13] T.D.C. Mattos, F.M. Santoro, K. Revoredo and V.T. Nunes, “Formalizing the Situation of a Business Process Activity,” in Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference on, pp. 128–134, 2012. Available: http://dx.doi.org/10.1109/cscwd.2012.6221808

[14] C. Stary, “Open Organizational Learning: Stakeholder Knowledge for Process Development,” in Knowledge Management & E-Learning: An International Journal (KM&EL), vol. 8, no. 1, pp. 86–108, 2016.

[15] F. Strecker, R. Gniza, T. Hollosy and F. Schmatzer, “Business-Actors as Base Components of Complex and Distributed Software Applications,” in Proceedings of the 8th International Conference on Subject-oriented Business Process Management, Erlangen, Germany, p. 8, ACM, 2016. Available: http://dx.doi.org/10.1145/2882879.2882887

[16] A.R. Silva, R. Meziani, R. Magalhaes, D. Martinho, A. Aguiar and N. Flores, “AGILIPO: Embedding Social Software Features into Business Pprocess Tools,” in Business Process Management Workshops, Springer Berlin Heidelberg, pp. 219–230, 2009.

[17] M.E. Rangiha, M. Comuzzi and B. Karakostas, “Role and Task Recommendation and Social Tagging to Enable Social Business Process Management,” in Enterprise, Business-Process and Information Systems Modeling LNBIP 214, Springer International Publishing, pp. 68–82, 2015. Available: http://dx.doi.org/10.1007/978-3-319-19237-6_5

[18] N. Argyropoulos, H. Mouratidis and A. Fish, “Towards the Derivation of Secure Business Process Designs,” in Advances in Conceptual Modeling, Springer International Publishing, pp. 248–258, 2015. Available: http://dx.doi.org/10.1007/978-3-319-25747-1_25

[19] M. Trkman, J. Mendling and M. Krisper, “Using Business Process Models to Better Understand the Dependencies Among User Stories,” in Information and Software Technology, vol. 71, pp. 58–76, 2016. Available: http://dx.doi.org/10.1016/j.infsof.2015.10.006

[20] J. Recker, N. Safrudin and M. Rosemann, “How Novices Design Business Processes,” in Information Systems, vol. 37, no. 6, pp. 557–573, 2012. Available: http://dx.doi.org/10.1016/j.is.2011.07.001

[21] A.W. Scheer, “Architecture of Integrated Information Systems,” in Foundations of Enterprise Modeling, Springer Science & Business Media, Berlin, 2000. Available: http://dx.doi.org/10.1007/978-3-642-97389-5

[22] J. Pinggera, P. Soffer, D. Fahland, M. Weidlich, S. Zugal, B. Weber and J. Mendling, “Styles in Business Process Modeling: An Exploration and a Model,” Software & Systems Modeling, vol. 14, no. 3, pp. 1055–1080, 2015. Available: https://doi.org/10.1007/s10270-013-0349-1

[23] C. Hewitt, P. Bishop and R. Steiger, “A Universal Modular Aactor Formalism for Artificial Intelligence,” in Proceedings of the 3rd international joint conference on Artificial intelligence. Morgan Kaufmann Publishers Inc, pp. 235–245, 1973.

[24] V. Vernon, “Reactive Messaging Patterns with the Actor Model: Applications and Integration in Scala and Akka,” Addison-Wesley Professional, New York, 2015.

[25] A.H. ter Hofstede, W. van der Aalst, M. Adams and N. Russell, Eds., “Modern Business Process Automation: YAWL and its Support Environment,” Springer Science & Business Media, 2009. Available: http://dx.doi.org/10.1108/k.2010.06739fae.002

[26] F. Krenn, C. Stary, “Dynamic Switching of Perspectives on Business Processes,” B. Johansson, F. Vencovský, Eds., Joint Proceedings of the BIR 2016 Workshops and Doctoral Consortium co-located with 15th International Conference on Perspectives in Business Informatics Research (BIR 2016), Prague, Czech Republic, September 14–16, 2016. [Online]. Available: http://ceur-ws.org/Vol-1684/paper13.pdf