Johannes Manner
Stephan Haarmann

ZEUS 2020

12th ZEUS Workshop, ZEUS 2020,
Potsdam, Germany, 20–21 February 2020
Proceedings

Volume Editors

Johannes Manner
University of Bamberg, Distributed Systems Group
An der Weberei 5, DE-96049 Bamberg
johannes.manner@uni-bamberg.de

Stephan Haarmann
Hasso Plattner Institute, Business Process Technology
Prof.-Dr.-Helmert-Str. 2-3, DE-14482 Potsdam
stephan.haarmann@hpi.de

Preface

In February 2020, we had the pleasure to organize the 12^{th} edition of the ZEUS Workshop in Potsdam, Germany. This workshop series offers young researchers an opportunity to present and discuss early ideas and work in progress as well as to establish contacts among young researchers. For this year's edition, we selected five regular submissions, three position papers, and one tool demonstration for presentation at the workshop. Each submission went through a thorough peer-review process and was assessed by at least five members of the program committee with regard to its relevance and scientific quality. The accepted contributions cover the areas of Business Process Management, Cloud Computing, Microservices, Software Design, and the Internet of Things.

The workshop program was further enriched by keynotes from both academia and the industry. Niall Deehan from Camunda Services GmbH gave a talk about Microservices with the title *Common Pitfalls in Microservice Integration and how to Avoid Them*. Prof. Dr. Florian Tschorsch, TU Berlin, presented his research on *Distributed Security Infrastructures* with the title *The Dream of Anonymous Data*.

The workshop was generously sponsored by Camunda Services GmbH.

Potsdam, February 2020

Johannes Manner
Stephan Haarmann

Organization

Steering Committee

Nico Herzberg	SAP SE
Oliver Kopp	Daimler AG
Stefan Kolb	University of Bamberg
Stephan Haarmann	Hasso Plattner Institute, University of Potsdam
Johannes Manner	University of Bamberg

Local Organizer

Stephan Haarmann	Hasso Plattner Institute, University of Potsdam

Program Committee Chairs

Nico Herzberg	SAP SE
Oliver Kopp	Daimler AG
Stefan Kolb	University of Bamberg
Stephan Haarmann	Hasso Plattner Institute, University of Potsdam
Johannes Manner	University of Bamberg

Program Committee

Saimir Bala	Vienna University of Economics and Business
Felix Baumann	PricewaterhouseCoopers GmbH WPG
Justus Bogner	University of Applied Sciences Reutlingen
Michael Borkowski	TU Wien
Achim D. Bruckner	University of Exeter
Sebastian Böhm	University of Bamberg
Linus W. Dietz	Technical University of Munich
Dirk Fahland	Eindhoven University of Technology
Matthias Geiger	University of Bamberg
Georg Grossmann	University of South Australia
Simon Harrer	innoQ Deutschland GmbH
Lukas Harzenetter	University of Stuttgart
Thomas Heinze	German Aerospace Center
Pascal Hirmer	University of Stuttgart
Christoph Hochreiner	Compass Verlag
Sven Ihde	Hasso Plattner Institute, University of Potsdam
Jan Ladleif	Hasso Plattner Institute, University of Potsdam
Jörg Lenhard	SAP SE
Henrik Leopold	Kühne Logistics University
Robin Lichtenthäler	University of Bamberg
Daniel Lübke	Digital Solution Architecture
Sankalita Mandal	HPI, University of Potsdam
Matteo Nardelli	University of Rome Tor Vergata
Adriatik Nikaj	Hasso Plattner Institute, University of Potsdam
Luise Pufahl	TU Berlin
Joel Scheuner	Chalmers | University of Gothenburg
Stefan Schulte	Vienna University of Technology
Christian Sturm	University of Bayreuth
Jan Sürmeli	FZI Forschungszentrum Informatik, Karlsruhe
Stefan Winzinger	University of Bamberg
Michael Wurster	University of Stuttgart
Tammo van Lessen	innoQ Deutschland GmbH
Han van der Aa	Humboldt University of Berlin

Sponsoring Institutions

Camunda Services GmbH

Table of Contents

Auto-scaling Policies to Adapt the Application Deployment in Kubernetes

Fabiana Rossi

Department of Civil Engineering and Computer Science Engineering
University of Rome Tor Vergata, Italy
f.rossi@ing.uniroma2.it

Abstract The ever increasing diffusion of computing devices enables a new generation of containerized applications that operate in a distributed cloud environment. Moreover, the dynamism of working conditions calls for an elastic application deployment, which can adapt to changing workloads. Despite this, most of the existing orchestration tools, such as Kubernetes, include best-effort threshold-based scaling policies whose tuning could be cumbersome and application dependent. In this paper, we compare the default threshold-based scaling policy of Kubernetes against our model-based reinforcement learning policy. Our solution learns a suitable scaling policy from the experience so to meet Quality of Service requirements expressed in terms of average response time. Using prototype-based experiments, we show the benefits and flexibility of our reinforcement learning policy with respect to the default Kubernetes scaling solution.

Keywords: Kubernetes · Elasticity · Reinforcement Learning · Self-adaptive systems.

1 Introduction

Elasticity allows to adapt the application deployment at run-time in face of changing working conditions (e.g., incoming workload) and to meet stringent Quality of Service (QoS) requirements. Exploiting operating system level virtualization, *software containers* allow to simplify the deployment and management of applications, also offering a reduced computational overhead with respect to virtual machines. The most popular container management system is Docker. It allows to simplify the creation, distribution, and execution of applications inside containers. Although the container management systems can be used to deploy simple containers, managing a complex application (or multiple applications) at run-time requires an *orchestration tool*. The latter automates container provisioning, management, communication, and fault-tolerance. Although several orchestration tools exist [5,8], *Kubernetes*[1], an open-source platform introduced by Google in 2014, is the most popular solution. Kubernetes includes a Horizontal

[1] https://kubernetes.io

J. Manner, S. Haarmann (Eds.): 12$^{\text{th}}$ ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

Pod Autoscaler enabling to automatically scale the application deployment using a threshold-based policy based on cluster-level metrics (i.e., CPU utilization). However, this threshold-based scaling policy is not well suited to satisfy QoS requirements of latency-sensitive applications. Determining a suitable threshold is cumbersome, requiring to identify the relation between a system metric (i.e., utilization) and an application metric (i.e., response time), as well as to know the application bottleneck (e.g., in terms of CPU or memory). In this paper, we compare the default threshold-based scaling policy of Kubernetes against model-free and model-based reinforcement learning policies [14]. Our model-based solution automatically learns a suitable scaling policy from the experience so to meet QoS requirements expressed in term of average response time. To perform such comparison, we use our extension of Kubernetes, which includes a more flexible autoscaler that can be easily equipped with new scaling policies. The remainder of the paper is organized as follows. In Section 2, we discuss related works. In Section 3, we describe the Kubernetes features. Then, we propose a reinforcement learning-based scaling policy to adapt at run-time the deployment of containerized applications (Section 4). In Section 5, we evaluate the proposed solutions using prototype-based experiments. We show the flexibility and efficacy of using a reinforcement learning solution compared to the default Kubernetes scaling policy. In Section 6, we outline the ongoing and future research directions.

2 Related Work

The elasticity of containers is carried out in order to achieve different objectives: to improve application performance (e.g., [4]), load balancing and resource utilization (e.g., [1,11]), energy efficiency (e.g., [3]), and to reduce the deployment cost (e.g., [6,2]). Few works also consider a combination of deployment goals (e.g., [18]). Threshold-based policies are the most popular approaches to scale containers at run-time (e.g., [4,10]). Also the noteworthy orchestration tools (e.g., Kubernetes, Docker Swarm, Amazon ECS, and Apache Hadoop YARN) usually rely on best-effort threshold-based scaling policies based on some cluster-level metrics (e.g., CPU utilization). However, all these approaches require a non-trivial manual tuning of the thresholds, which can also be application-dependent. To overcome to this issue, solutions in literature propose container deployment methods ranging from mathematical programming to machine learning solutions. The mathematical programming approaches exploit methods from operational research in order to solve the application deployment problem (e.g., [12,13,18]). Since such a problem is NP-hard, other efficient solutions are needed. In the last few years, reinforcement learning (RL) has become a widespread approach to solve the application deployment problem at run-time. RL is a machine learning technique by which an agent can learn how to make (scaling) decisions through a sequence of interactions with the environment [15]. Most of the existing solutions consider the classic model-free RL algorithms (e.g., [7,16,17]), which however suffer from slow convergence rate. To tackle this issue, in [14], we propose a novel model-based RL solution that exploits what is known (or can be estimated) about the system dynamics to adapt the application deployment at run-time. Experimental results based on Docker Swarm have shown the flexibility of our approach, which can learn different adaptation strategies according to the optimized deployment objectives (e.g., meet QoS requirements in terms of average response time). Moreover, we have shown that the model-based RL agent learns a better adaptation policy than other model-free RL solutions. Encouraged by the previous promising results, in this paper, we integrate the model-based RL solution in Kubernetes, one of the most popular container orchestration tools used in the academic and industrial world. Experimental results in [8] demonstrate that Kubernetes performs better than other existing orchestration tools, such as Docker Swarm, Apache Mesos, and Cattle. However, Kubernetes is not suitable for managing latency-sensitive applications in a extremely dynamic environment. It is equipped with a static best-effort deployment policy that relies on system-oriented metrics to scale applications in face of workload variations. In this paper, we first extend Kubernetes to easily introduce self-adaptation capabilities. Then, we integrate RL policies in Kubernetes and compare them against the default Kubernetes auto-scaling solution.

3 Kubernetes

Kubernetes is an open-source orchestration platform that simplifies the deployment, management, and execution of containerized applications. Based on a master-worker decentralization pattern, it can replicate containers for improving resource usage, load distribution, and fault-tolerance. The master node maintains the desired state at run-time by orchestrating applications (using pods). A worker is a computing node that offers its computational capability to enable the execution of pods in distributed manner. A pod is the smallest deployment unit in Kubernetes. When multiple containers run within a pod, they are co-located and scaled as an atomic entity. To simplify the deployment of applications, Kubernetes introduces Deployment Controllers that can dynamically create and destroy pods, so to ensure that the desired state (described in the *deployment* file) is preserved at run-time. Kubernetes also includes a Horizontal Pod Autoscaler[2] to automatically scale the number of pods in a Deployment based on the ratio between the target value and the observed value of pod's CPU utilization. Setting the CPU utilization threshold is a cumbersome and error-prone task and may require a knowledge of the application resource usage to be effective.

To address this limitation, we equip Kubernetes with a decentralized control loop. In a single loop iteration, it monitors the environment and the containerized applications, analyzes application-level (i.e., response time) and cluster-level (i.e., CPU utilization) metrics, and plans and executes the corresponding scaling actions. The modularity of the control loop allows us to easily equip it with different QoS-aware scaling policies. To dynamically adapt the application deployment according to the workload variations, we consider RL policies.

---

[2] `https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/`

4 Reinforcement Learning Scaling Policy

Differently from the Kubernetes scaling policy, we aim to design a flexible solution that can be easily customized by manually tuning various configuration parameters. In this paper, we customize the RL solution proposed in [14] to scale at run-time the number of application instances (i.e., pods). RL refers to a collection of trial-and-error methods by which an agent must prefer actions that it found to be effective in the past (*exploitation*). However, to discover such actions, it has to explore new actions (*exploration*). In a single control loop iteration, the RL agent selects the adaptation action to be performed. As first step, according to the received application and cluster-oriented metrics, the RL agent determines the Deployment Controller state and updates the expected long-term cost (i.e., Q-function). We define the application state as $s = (k, u)$, where k is the number of application instances (i.e., pods), and u is the monitored CPU utilization. We denote by $\mathcal{S}$ the set of all the application states. We assume that $k \in \{1, 2, ..., K_{\max}\}$; being the CPU utilization (u) a real number, we discretize it by defining that $u \in \{0, \bar{u}, ..., L\bar{u}\}$, where $\bar{u}$ is a suitable quanta. For each state $s \in \mathcal{S}$, we define the set of possible adaptation actions as $\mathcal{A}(s) \subseteq \{-1, 0, 1\}$, where ± 1 defines a scaling action (i.e., $+1$ to scale-out and -1 to scale-in), and 0 is the *do nothing* decision. Obviously, not all the actions are available in any application state, due to the upper and lower bounds on the number of pods per application (i.e., $K_{\max}$ and 1, respectively). Then, according to an action selection policy, the RL agent identifies the scaling action a to be performed in state s. The execution of a in s leads to the transition in a new application state (i.e., s') and to the payment of an immediate cost. We define the immediate cost $c(s, a, s')$ as the weighted sum of different terms, such as the *performance penalty*, c_{perf}, *resource cost*, c_{res}, and *adaptation cost*, c_{adp}. We normalized them in the interval $[0, 1]$, where 0 represents the best value (no cost), 1 the worst value (highest cost). Formally, we have: $c(s, a, s') = w_{\text{perf}} \cdot c_{\text{perf}} + w_{\text{res}} \cdot c_{\text{res}} + w_{\text{adp}} \cdot c_{\text{adp}}$, where w_{adp}, w_{perf} and w_{res}, $w_{\text{adp}} + w_{\text{perf}} + w_{\text{res}} = 1$, are non negative weights that allow us to express the relative importance of each cost term. We can observe that the formulation of the immediate cost function $c(s, a, s')$ is general enough and can be easily customized with other QoS requirements. The *performance penalty* is paid whenever the average application response time exceeds the target value $R_{\max}$. The *resource cost* is proportional to the number of application instances (i.e., pods). The *adaptation cost* captures the cost introduced by Kubernetes to perform a scaling operation. The traffic routing strategy used in Kubernetes forwards the application requests to the newly added pod, even if not all containers in the pod are already running. We observe that, for this reason, we prefer horizontal scaling to vertical scaling operations. When a vertical scaling changes a pod configuration (e.g., to update its CPU limit), Kubernetes spawns new pods as a replacement of those with the old configuration. In this phase, the application availability decreases and only a subset of the incoming requests are processed. Conversely, a scale-out action introduces a reduced adaptation cost inversely proportional to the number of application instances.

The received immediate cost contributes to update the Q-function. The Q-function consists in $Q(s,a)$ terms, which represent the expected long-term cost that follows the execution of action a in state s. The existing RL policies differ in how they update the Q-function and select the adaptation action to be performed (i.e., action selection policy) [15]. To adapt the application deployment, we consider a model-based solution which we have extensively evaluated in [14]. At any decision step, the proposed model-based RL solution does not use an action selection policy (e.g., ϵ-greedy action selection policy) but it always selects the best action in term of Q-values, i.e., $a = \arg\min_{a' \in A(s)} Q(s,a')$. Moreover, to update the Q-function, the simple weighted average of the traditional RL solutions (e.g., Q-learning) is replaced by the Bellman equation [15]:

$$Q(s,a) = \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[c(s,a,s') + \gamma \min_{a' \in \mathcal{A}} Q(s',a') \right] \quad \substack{\forall s \in \mathcal{S}, \\ \forall a \in \mathcal{A}(s)} \tag{1}$$

where $\gamma \in [0,1)$ is the *discount factor*, $p(s'|s,a)$ and $c(s,a,s')$ are, respectively, the transition probabilities and the cost function $\forall s, s' \in \mathcal{S}$ and $a \in A(s)$. Thanks to the experience, the proposed model-based solution is able to maintain an empirical model of the unknown external system dynamics (i.e., $p(s'|s,a)$ and $c(s,a,s')$) speeding-up the learning phase. Further details on our model-based RL solution can be found in [14].

5 Results

We show the self-adaptation capabilities of Kubernetes when equipped with model-free and model-based RL policies as well as the default threshold-based solution (by the Horizontal Pod Autoscaler). The RL solutions scale pods using user-oriented QoS attributes (i.e., response time), whereas the Horizontal Pod Autoscaler uses a best-effort threshold-based policy based on cluster-level metrics (i.e., CPU utilization). The evaluation uses a cluster of 4 virtual machines of the Google Cloud Platform; each virtual machine has 2 vCPUs and 7.5 GB of RAM (type: n1-standard-2). We consider a reference CPU-intensive application that computes the sum of the first n elements of the Fibonacci sequence. As shown in Figure 1, the application receives a varying number of requests. It follows the workload of a real distributed application [9], accordingly amplified

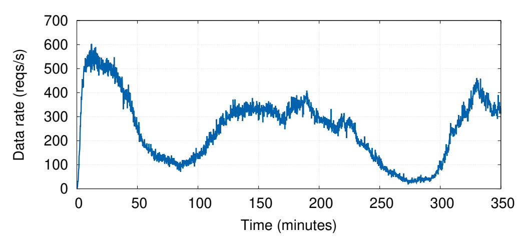
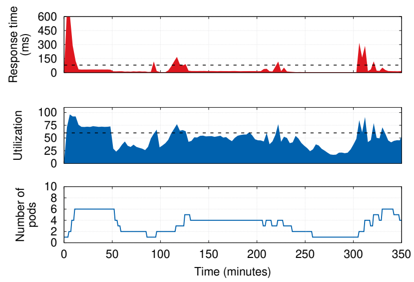
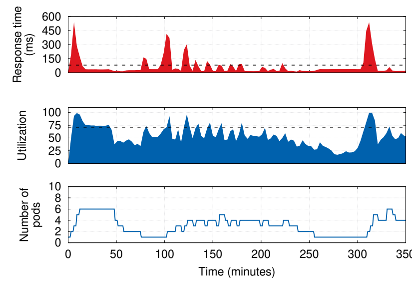


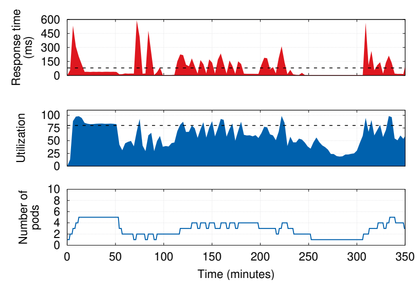
Figure 1: Workload used for the reference application.



(a) Threshold at 60% of CPU utilization.



(b) Threshold at 70% of CPU utilization.

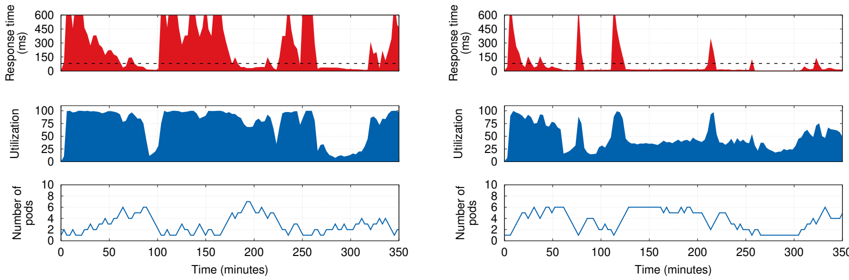


(c) Threshold at 80% of CPU utilization.

Figure 2: Application performance using Horizontal Pod Autoscaler.

and accelerated so to further stress the application resource requirements. The application expresses the QoS in terms of target response time $R_{\max} = 80$ ms. To meet $R_{\max}$, it is important to accordingly adapt the number of application instances. The Kubernetes autoscaler executes a control loop every 3 minutes. To learn an adaptation policy, we parameterize the model-based RL algorithm as in our previous work [14]. For sake of comparison, we consider also the model-free Q-learning approach that chooses a scaling action according to the ϵ-greedy selection policy: at any decision step, the Q-learning agent chooses, with probability ϵ, a random action, whereas, with probability $1-\epsilon$, it chooses the best known action. For Q-learning, we set ϵ to 10%. To discretize the application state, we use $K_{\max} = 10$ and $\bar{u} = 0.1$. For the immediate cost function, we consider the set of weights $w_{\text{perf}} = 0.90$, $w_{\text{res}} = 0.09$, $w_{\text{adp}} = 0.01$. This weight configuration allows to optimize the application response time, considered to be more important than saving resources and reducing the adaptation costs.

The default Kubernetes threshold-based scaling policy is application unaware and not flexible, meaning that it is not easy to satisfy QoS requirements of latency-sensitive applications by setting a threshold on CPU utilization (see Figures 2a–2c). From Table 1, we can observe that small changes in the threshold setting lead to a significant performance deterioration. Setting the scaling



(a) Q-learning. (b) Model-based RL.

Figure 3: Application performance using RL policies.

Table 1: Application performance under the different scaling policies.

Elasticity Policy	$R_{\max}$ *violations* (%)	*Average CPU utilization* (%)	*Average number of pods*	*Median of Response time (ms)*	*Adaptations* (%)
Model-based	14.40	48.51	3.75	16.38	56.00
Q-learning	64.0	76.94	2.90	201.71	65.6
HPA thr = 60	9.20	50.81	3.54	16.11	6.38
HPA thr = 70	21.43	54.43	3.14	34.61	10.96
HPA thr = 80	40.12	63.70	3.18	37.54	12.89

threshold is cumbersome, e.g., with threshold on 80% of CPU utilization, we obtain a rather high number of $R_{\max}$ violations. With the scaling threshold at 70% of CPU utilization, the application violates $R_{\max}$ 21% of time, with 54% of average CPU utilization. With the scaling threshold at 60% of CPU utilization, the application has better performance ($R_{\max}$ is exceeded only 9% of time), even though we might still perform a finer threshold tuning to further increase it.

Conversely, the RL approach is general and more flexible, requiring only to specify the desired deployment objectives. It allows to indicate *what* the user aims to obtain (through the cost function weights), instead of *how* it should be obtained. In particular, a RL learning agent learns the scaling policy in an automatic manner. Figures 3a and 3b show the application performance when the model-free and model-based RL solutions are used. The RL agent starts with no knowledge on the adaptation policy, so it begins to explore the cost of each adaptation action. When Q-learning is used, the RL agent slowly learns how to adapt the application deployment. As we can see from Figure 3a and Table 1, the application deployment is continuously updated (i.e., 66% of the time) and the RL agent does not learn a good adaptation policy within the experiment duration. As a consequence, the application response time exceeds $R_{\max}$ most of the time. Taking advantage of the system knowledge, the model-based solution has a very different behavior: it obtains better performance and more quickly reacts to workload variations. We can see that, in the first minutes of the experiment, the model-based solution does not always respect the target application response time. However, as soon as a suitable adaptation policy is learned, the model-based RL solution can successfully scale the application and meet the application response time requirement most of the time. The learned adaptation policy deploys a number of pods that follows the application workload (see Figures 1 and 3b), maintaining a reduced number of $R_{\max}$ violations (14.4%) and a good average resource utilization (49%).

We should observe that, even though a fine grained threshold tuning can be performed (thus improving performance of the default Kubernetes scaling policy), the RL-based approach automatically learns a suitable and satisfying adaptation strategy. Moreover, changing the cost function weights, the RL solution can easily learn different scaling policies, e.g., to improve resource utilization or to reduce deployment adaptations [14].

6 Conclusion

Kubernetes is one of the most popular orchestration tools to manage containers in a distributed environment. To react to workload variations, it includes a threshold-based scaling policy that changes the application deployment according to cluster-level metrics. However, this approach is not well-suited to meet stringent QoS requirements. In this paper, we compare model-free and model-based RL scaling policies against the default threshold-based solution. The prototype-based results have shown the flexibility and benefits of RL solutions: while the model-free Q-learning suffers from slow convergence time, the model-based approach can successfully learn the best adaptation policy, according to the user-defined deployment goals.

As future work, we plan to investigate the deployment of applications in geo-distributed environment, including edge/fog computing resources located at the network edges. The default Kubernetes scheduler spreads containers on computing resources not taking into account the not-negligible network delay among them. This can negatively impact the performance of latency-sensitive applications. Therefore, alongside the elasticity problem, also the placement problem (or scheduling problem) should be efficiently solved at run-time. We want to extend the proposed heuristic so to efficiently control the scaling and placement of multi-component applications (e.g., micro-services). When an application consists of multiple components that cooperate to accomplish a common task, adapting the deployment of a component impacts on performance of the other components. We are interested in considering the application as a whole, so to develop policies that can adapt, in proactive manner, the deployment of inter-connected components, avoiding performance penalties.

Acknowledgment

The author would like to thank her supervisor Prof. Valeria Cardellini and to acknowledge the support by Google with the GCP research credits program.

References

1. Abdelbaky, M., Diaz-Montes, J., Parashar, M., Unuvar, M., Steinder, M.: Docker containers across multiple clouds and data centers. In: Proc. of IEEE/ACM UCC 2015. pp. 368–371 (2015)
2. Al-Dhuraibi, Y., Paraiso, F., Djarallah, N., Merle, P.: Autonomic vertical elasticity of Docker containers with ElasticDocker. In: Proc. of IEEE CLOUD '17. pp. 472–479 (2017)
3. Asnaghi, A., Ferroni, M., Santambrogio, M.D.: DockerCap: A software-level power capping orchestrator for Docker containers. In: Proc. of IEEE EUC '16. pp. 90–97 (2016)
4. Barna, C., Khazaei, H., Fokaefs, M., Litoiu, M.: Delivering elastic containerized cloud applications to enable DevOps. In: Proc. of SEAMS '17. pp. 65–75 (2017)
5. Casalicchio, E.: Container orchestration: A survey. In: Systems Modeling: Methodologies and Tools, pp. 221–235. Springer International Publishing, Cham (2019)
6. Guan, X., Wan, X., Choi, B.Y., Song, S., Zhu, J.: Application oriented dynamic resource allocation for data centers using Docker containers. IEEE Commun. Lett. **21**(3), 504–507 (2017)
7. Horovitz, S., Arian, Y.: Efficient cloud auto-scaling with SLA objective using Q-learning. In: Proc. of IEEE FiCloud '18. pp. 85–92 (2018)
8. Jawarneh, I.M.A., Bellavista, P., Bosi, F., Foschini, L., Martuscelli, G., Montanari, R., Palopoli, A.: Container orchestration engines: A thorough functional and performance comparison. In: Proc. of IEEE ICC 2019. pp. 1–6 (2019)
9. Jerzak, Z., Ziekow, H.: The DEBS 2015 grand challenge. In: Proc. ACM DEBS 2015. pp. 266–268 (2015)
10. Khazaei, H., Ravichandiran, R., Park, B., Bannazadeh, H., Tizghadam, A., Leon-Garcia, A.: Elascale: Autoscaling and monitoring as a service. In: Proc. of CASCON '17. pp. 234–240 (2017)
11. Mao, Y., Oak, J., Pompili, A., Beer, D., Han, T., Hu, P.: DRAPS: dynamic and resource-aware placement scheme for Docker containers in a heterogeneous cluster. In: Proc. of IEEE IPCCC '17. pp. 1–8 (2017)
12. Nardelli, M., Cardellini, V., Casalicchio, E.: Multi-level elastic deployment of containerized applications in geo-distributed environments. In: Proc. of IEEE FiCloud '18. pp. 1–8 (2018)
13. Rossi, F., Cardellini, V., Lo Presti, F.: Elastic deployment of software containers in geo-distributed computing environments. In: Proc. of IEEE ISCC '19. pp. 1–7 (2019)
14. Rossi, F., Nardelli, M., Cardellini, V.: Horizontal and vertical scaling of container-based applications using Reinforcement Learning. In: Proc. of IEEE CLOUD '19. pp. 329–338 (2019)
15. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, 2 edn. (2018)
16. Tang, Z., Zhou, X., Zhang, F., Jia, W., Zhao, W.: Migration modeling and learning algorithms for containers in fog computing. IEEE Trans. Serv. Comput. **12**(5), 712–725 (2019)
17. Tesauro, G., Jong, N.K., Das, R., Bennani, M.N.: A hybrid Reinforcement Learning approach to autonomic resource allocation. In: Proc. of IEEE ICAC '06. pp. 65–73 (2006)
18. Zhao, D., Mohamed, M., Ludwig, H.: Locality-aware scheduling for containers in cloud computing. IEEE Trans. Cloud Comput. (2018)

Incorporating Organizational Aspects into Fragment-based Case Management

Simon Remy

Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
simon.remy@hpi.de

Abstract. Business process management (BPM) enables organizations to model and analyze their business processes, for example, with the help of the Business Process Model and Notation (BPMN). Concerning knowledge-intensive and flexible processes, recent research identified a gap between implemented processes and modeled ones. To close this gap, several approaches have been developed. One of them is fragment-based case management (fCM). However, these approaches share a data-centric view on processes. This work presents an approach to enrich process fragments with organizational aspects. For this purpose, a meta-model that describes the utilization of process participants in fragment-based case management will be introduced. Further, it will be demonstrated how to apply the approach to BPMN models to derive organizational aware process fragments.

Keywords: Business Process Management · Fragment-based Case Management · Roles.

1 Introduction

As an interdisciplinary research field between computer science and business administration, *business process management* (BPM) enables organizations to design, administrate, configure, and analyze their processes [17]. Since the outcome of most business processes is the result of the execution of subsequent activities, BPM provides methods to analyze and to understand the relationships between them [17]. One way to represent these relations and interactions are process models using the *Business Process Model and Notation* (BPMN) standard. Among others, BPMN provides basic elements to model activities, events, and control-flow. Further, BPMN aims to close the gap between process modeling and implementation [14]. Because of this, process models can also be executed using process engines.

Lately, BPM has been applied in many different enterprises and industries. However, it became clear that there exists a gap between some real-life business processes and initially modeled ones. Many processes require a certain amount of flexibility and are limited by traditional workflow management systems [3]. Especially knowledge-intensive processes are affected by this, like treatment processes in healthcare. Those processes are rather unstructured compared to e.g.,

J. Manner, S. Haarmann (Eds.): 12th ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

a production process [6]. To better support knowledge-intensive processes, a new paradigm was introduced, namely case handling or also called case management [3, 11]. Based on this, other concepts like artifact centric models, *Guard Stage Milestone* models (GSM), *Adaptive Case Management* (ACM), and *Production Case Management* (PCM) were developed [5, 10, 12, 13].

These approaches have in common that they especially focus on the data perspective. The states of data-objects indicate the state of a case and enable knowledge-workers to make decisions about future steps in a case. However, none of the approaches explicitly incorporate organizational perspectives, like roles.

In this paper, we will present a meta-model for process fragments, which can be used in *fragment-based Case Management* (fCM), a specification of PCM. Besides data objects, the model considers the organizational perspective of business processes. Since knowledge-workers play an essential role in such processes, we aim to provide a way to explicitly include them in the modeling process as well as the relations between them. Further, we will demonstrate how a standard BPMN model can be transformed into role-specific fragments.

The remainder of this paper is organized as follows. In section 2, we introduce a running example, followed by related work in section 3. Section 4 presents a meta-model to formally describe the usage of roles in process fragments and a demonstration of deriving fragments from BPMN models based on it. Results, limitations, and future work are discussed in section 5.

2 Running Example

Figure 1 depicts a sample BPMN process model. The model shows a simplified treatment process of patients visiting the cardiology ward of a hospital. While the patient is only modeled implicit via the activity labels, the model consists of three lanes: nurse, physician, and both. While the BPMN standard does not specify the usage of lanes [14], they are commonly used to assign activities to a specific resource, in this example, roles.

Whenever a new patient enters the ward, a new process instance will be instantiated. First, a nurse admits the patient, collects her medical history and updates the patient's record. Next, a blood sample is drawn from the patient. This can either be done by the nurse or the physician, depending on who is available. After that, both have to examine the patient together, followed by an activity to prescribe a treatment by the physician. Before the process ends, the nurse releases the patient and updates the patient's record concurrently.

While executing activities, data objects will be read and written. Changes to them are indicated by changes in the data objects' state, e.g., after the patient has been admitted the state of the *Patient* data object changes from *init* to *admitted*. The state-space of each data object is defined by its lifecycle, which is not part of the process model (see [11] for details). While data objects are bound to specific process instances, they do not only persist information during its lifetime but also define *InputSets* and *OutputSets* of activities. Those sets can be seen as preconditions and postconditions of the respective activities. In other

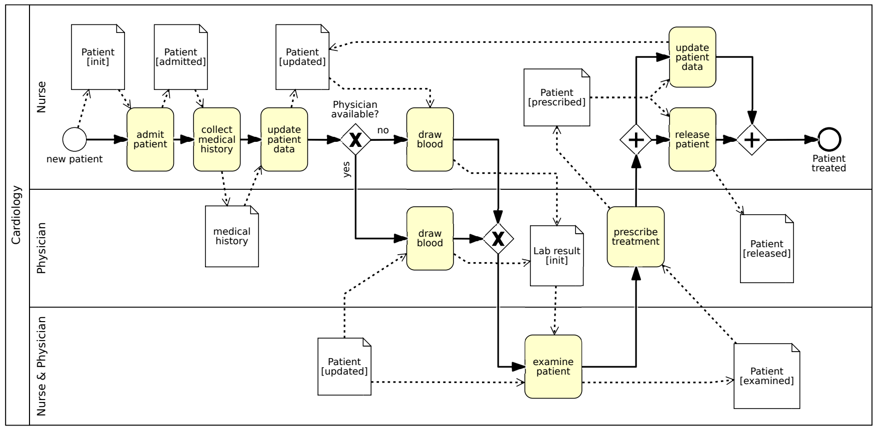


Fig. 1. Sample process that depicts the treatment process of patients in a cardiology department of a hospital. Indicated by the lanes, participants of two roles are involved. However, one activity requires participants of both roles to be executed.

words, an activity can be control-flow enabled, but not data-flow enabled. This is the case if not all data objects in its *InputSet* are in the required state [11].

Even if the presented example depicts a simple process, and therefore the use of fCM is limited, our findings can be applied to more complex models. We will use the example to illustrate how to derive process fragments for business processes based on roles according to the meta-model, described in section 4. For the remainder, we lift the assumption, that only explicitly modeled roles, like lanes in process models, will be considered.

3 Related Work

As described in the previous section, traditional process management approaches are limited in their capability to support knowledge-intensive processes. To overcome these limitations, several approaches have been proposed in the past.

As one of the first approaches case handling has been developed [3]. This approach not only focuses on the order of activities but mainly on data-objects. Since then, several other ideas were introduced. Business artifact centric approaches focus on the life-cycles of business objects to describe the context and structure of processes [13]. GSM follows another data-driven approach using guards, stages, and milestones to structure processes [5]. With the *Case Management Model and Notation* (CMMN), a new modeling standard has been introduced to support case management [15]. Process fragments were introduced by PCM to model small parts of a process to maintain a certain degree of the structure without limiting its flexibility too much [4, 11, 12]. Lastly, ACM aims to enable knowledge-worker to adopt processes at run-time[10].

In their literature review, Hauder et al. identify several research questions for ACM[8], where some can also be applied to the previously presented approaches. According to their work, successful case management requires collaborations between different roles and clear rules for interactions [9, 16]. Further, the authors understand roles as a powerful tool to restrict data access and to ensure data privacy [8]. One approach to model communication and interactions in business processes is the *Design & Engineering Methodology for Organizations* (DEMO) [7]. Other approaches are data-driven and based on historical process data. They aim to model social networks from recorded process data. Those networks visualize interactions between process participants and roles [1, 2].

4 Organizational Perspective on Process Fragments

To provide a formal basis, to discuss the usage of roles in fCM, we introduce a meta-model for process fragments in the following. Further, we show an application of the approach to derive fragments from BPMN process models.

4.1 Meta-Model

The meta-model, depicted in Figure 2, is based on the definition of process models, presented in [17]. *Process Fragment* is the central class of the meta-model. A fragment consists of *Edges*, *Nodes*, *Data Objects*, and *Roles*, where each edge connects exactly two nodes. However, nodes can be connected to multiple edges. Therefore edges express the control-flow relationship between nodes.

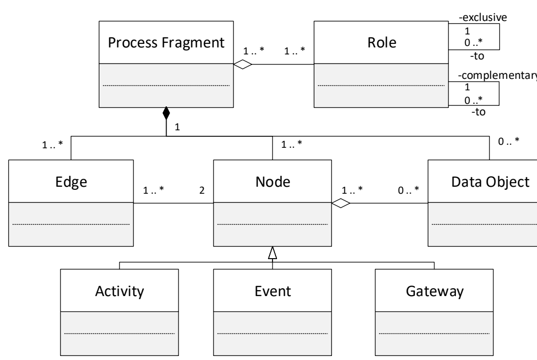


Fig. 2. Meta-model for process fragments

Further, a *Node* can be an *Activity*, an *Event*, or a *Gateway*. In difference to the process meta-model, a fragment can consist of a single activity. Since fragments must not have empty start events, a fragment is considered as enabled if all preconditions of the first activity are fulfilled, in other words, as soon as it is dataflow-enabled [11].

Data Objects play an essential role in fCM and are therefore included in the meta-model. Multiple data objects can be associated with a set of nodes. However, not every node has to be associated with a data object. Thus, a fragment does not require data objects at all. While this seems to contradict the purpose of fCM, it allows the process modeler, to design process fragments, which are enabled at any time during the execution of an instance, like escalating the case to a higher level, e.g. a manager.

Lastly, each process fragment is associated with at least one *Role*. Only participants, who belong to the associated roles are able to execute instances of the fragment. However, multiple roles can be associated with the same fragment. In this case, roles can be either mutual exclusive to each other or complementary. In the first case, only participants of one role can be involved during run-time, while in the second case, participants of all roles have to participate in its execution.

4.2 Application

The first step to derive fragments is to split the process model horizontally based on each lane. As a result, activities in one lane will be disconnected whenever a handover between two lanes takes place. Those disconnected activity sequences are *fragment candidates*. However, fragments have to satisfy two conditions (*i*) be free of open *(X)OR-Joins/Splits*, and (*ii*) no shared activities with any other fragment. A join or split will be considered as open if its respective counterpart is not part of the same fragment.

In order to satisfy the first condition, the control-flow of a candidate will be cut before or after one of the respective gateways. In our running example, this is the case for the activity *draw blood* in the upper lane.This activity also violates the second condition, as it is part of an other fragment, that belongs to the *Physician*. After all fragment candidates satisfy the first condition, they have to be checked for shared activities. Depending on the control-flow structure, shared activities need to be handled differently. In the simplest case, a shared activity A is part of a sequence, without any exclusive and parallel gateways. In this case, the sequence is split into two or three fragments, depending on the position of A. If A belongs to a branch after an AND-Split, three scenarios, depending on the total number of branches and on the number of branches A belongs to, are possible. Given two branches, where A only belongs to one of them, a new fragment is created for the branch, that contains activity A. The other branch will be preserved as a sequence of the original fragment. If more branches exist and A still belongs to only one branch, only the effected branch needs to be removed, and a new fragment will be created. If activity A is part of multiple branches, a new fragment for each of them will be created. Depending on the number of not effected branches, the split can be preserved or not. Independent of the applicable scenario, all newly created fragments might need to be split up further in order to satisfy the second condition. Also, in order to keep the semantic of the AND-join, its conditions need to be reflected in the data-flow. Regarding the running example, the activity *update patient data* occurs in two fragments, and the first described scenario applies. Since the control-flow is only split into two concurrent branches, both will be transformed into a fragment.

Further, if activity A is part of an XOR/OR-Split, the following steps need to be performed. All branches that contain A will be removed, and for each, a new fragment will be created. If at least one branch does not contain A, a new edge from the split node, to the join node will be inserted. Again, all newly created fragments might need to be split up further in order to satisfy the second condition. After all fragments have been derived and comply with both conditions, the corresponding roles will be associated with the fragments. Since multiple roles can be associated with one fragment, logical expressions are used to express the relations between them. Roles, which are mutually exclusive to each other, are joined by the $\vee$ operator, while complimentary roles are connected using the $\wedge$ expression. Fragments that share the same set of roles, including the same relations, are grouped as a collection.

Figure 3 depicts six fragments that are derived from the process model presented in section 1. The graphical presentation of the fragments is loosely based on the BPMN standard. Single fragments are modeled using core BPMN elements, like activities, gateways, and events. The fragments are grouped based on their associated role, which is located in the upper left corner. If one collection is associated with multiple roles, all are listed, including their relation operator.

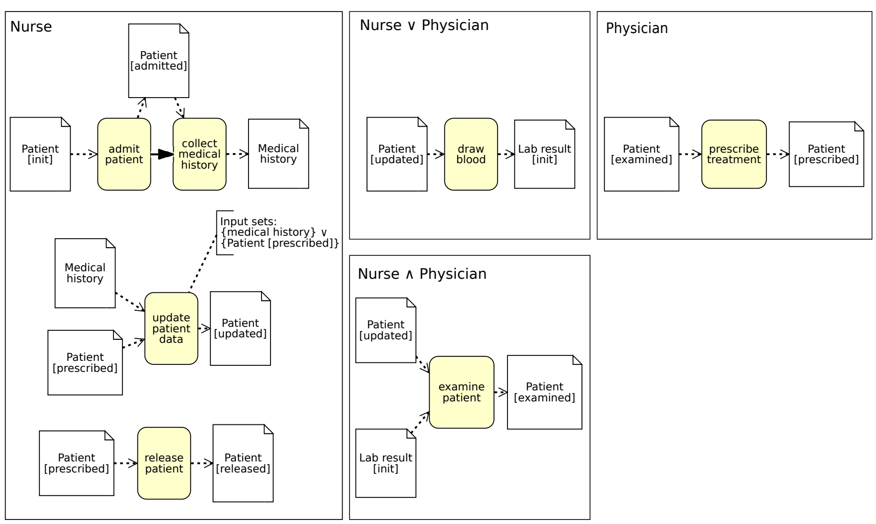


Fig. 3. Six fragments, derived from the BPMN model presented in section 1. The fragments are organized in collections according to their associated roles and their relations.

5 Discussion and Future Work

In this paper, we introduced a novel approach on how to integrate roles into process fragments to support fCM at design time. We presented a meta-model to define the components and provided a brief example of how to derive process fragments based on an existing process model.

Following this approach, adding new roles to an existing business process can easily be done by introducing a new collection of fragments or by adding role identifiers to existing ones, instead of editing a whole process model. Since fragments are organized in role-specific collections, it is also easier to remove them from the process. The compact representation provides a good overview in which parts of the process a role is involved and, therefore, where deadlocks or other inconsistencies may occur. This also goes along with better privacy protection, since fragments clearly show interactions between roles and data objects. In difference to BPMN, this approach provides a clear semantic of the relationships between roles. While in BPMN the behavior of, shared lanes or grouped activities, is not ultimately defined [14].

Like in BPMN, our approach does not specify any resource allocation method. Hence it would be possible that different participants of the same role are involved in different activities of the same fragment instance. Further, deriving process fragments from BPMN models can be challenging, regarding the semantic of the original control-flow. While the concurrent execution of multiple activities can be easily modeled using the respective BPMN elements, this is more complex with concurrent process fragments. The existing join condition has to be projected on the data-flow, using dedicated input sets of the subsequent fragments.

In this paper, we investigated a model-driven perspective to derive process fragments. In future work, we will explore a data-driven method based on event logs. Further, we will evaluate our approach based on real-life event logs concerning usability and interpretability.

References

1. van der Aalst, W.: Process Mining - Data Science in Action. Springer, Berlin, Heidelberg (2016)
2. van der Aalst, W., Reijers, H., Song, M.: Discovering social networks from event logs. Computer Supported Cooperative Work (CSCW) 14(6), 549–593 (2005)
3. van der Aalst, W., Weske, M., Grünbauer, D.: Case handling: a new paradigm for business process support. Data & Knowledge Engineering 53(2), 129–162 (2005)
4. Beck, H., Hewelt, M., Pufahl, L.: Extending fragment-based case management with state variables. In: Dumas, M., Fantinato, M. (eds.) Business Process Management Workshops, vol. 281, pp. 227–238. Springer International Publishing, Cham (2017)
5. Cohn, D., Hull, R.: Business artifacts: A data-centric approach to modeling business operations and processes. IEEE Data Engineering 32(3), 59 (2009)
6. Di Ciccio, C., Marrella, A., Russo, A.: Knowledge-intensive processes: Characteristics, requirements and analysis of contemporary approaches. Journal on Data Semantics 4(1), 29–57 (Mar 2015)

7. Dietz, J.L.G.: Designing technical systems as social systems. in: The language action perspective on communication modelling. In: Proceedings of the 8th International Working Conference on the Language-Action Perspective on Communication Modelling (LAP 2003), 2003 (2003)
8. Hauder, M., Pigat, S., Matthes, F.: Research challenges in adaptive case management: A literature review. In: 2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations. pp. 98–107. IEEE (2014)
9. Heil, S., Wild, S., Gaedke, M.: Collaborative adaptive case management with linked data. In: Proceedings of the 23rd International Conference on World Wide Web - WWW '14 Companion. pp. 99–102. ACM Press (2014)
10. Herrmann, C., Kurz, M.: Adaptive case management: Supporting knowledge intensive processes with IT systems. In: Schmidt, W. (ed.) S-BPM ONE - Learning by Doing - Doing by Learning, vol. 213, pp. 80–97. Springer Berlin Heidelberg (2011)
11. Hewelt, M., Weske, M.: A hybrid approach for flexible case modeling and execution. In: La Rosa, M., Loos, P., Pastor, O. (eds.) Business Process Management Forum, vol. 260, pp. 38–54. Springer International Publishing (2016)
12. Meyer, A., Herzberg, N., Puhlmann, F., Weske, M.: Implementation framework for production case management: Modeling and execution. In: 2014 IEEE 18th International Enterprise Distributed Object Computing Conference. pp. 190–199. IEEE (2014)
13. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems Journal 42(3), 428–445 (2003)
14. Object Management Group: Business process model and notation (bpmn) (2011), `https://www.omg.org/spec/BPMN/2.0/PDF`
15. Object Management Group: Case management model and notation (2016), `https://www.omg.org/spec/CMMN/1.1/PDF`
16. Reinhardt, W., Schmidt, B., Sloep, P., Drachsler, H.: Knowledge worker roles and actions—results of two empirical studies. Knowledge and Process Management 18(3), 150–174 (2011)
17. Weske, M.: Business Process Management - Concepts, Languages, Architectures. Springer-Verlag Berlin Heidelberg, 3 edn. (2019)

Managing Social Challenges in Cross-Organizational Event-Based Systems

Laura S. Thiele[1,2] and Nico Brehm[1]

[1] University of Applied Science Jena, Department of Industrial Engineering, Carl-Zeiß-Promenade 2, 07745 Jena, Germany
[2] German Aerospace Center (DLR), Institute of Data Science, Mälzerstraße 3, 07745 Jena, Germany
[laura.thiele,nico.brehm]@eah-jena.de

Abstract. During the last decade the manufacturing industry focused on the realization of industry 4.0 aspects. Besides the implementation of new technologies, existing software structures also need to be reviewed and adapted in this context. To stay competitive in the global market, especially small and medium-sized companies need to emphasize on better cooperation with other organizations. This leads to the implementation of cross-organizational distributed software system structures. The development of distributed systems faces different challenges – technical and code-centric as well as social challenges. This paper focuses on the social challenges that appear in distributed development processes. After defining the main challenges, the paper introduces a development approach that is based on the integration of a Federated Management System (FMS). FMS is a technical approach to minimize social challenges by the generation of system transparency and the provision of a platform for communication and interaction. It facilitates a distributed system development of cross-organizational event-based systems.

Keywords: Software Development · Distributed Systems · Event-Driven Systems · Wiki-Based System Management.

1 Introduction

The manufacturing industry is subject to current trends in the market, such as increasing product variety, custom and individual fabrication, as well as reducing production and delivery times [4, 12, 13]. Due to these trends, the manufacturing industry started to adapt traditional structures of their manufacturing in order to implement the ideas of Industry 4.0.

Most of the Industry 4.0 approaches are based on comprehensive data acquisition and usage. By now, data is captured by several different devices and systems. It needs to be collected and stored in order to process it for further purposes. To support the manufacturing industry in its current challenges, several research projects, e.g., [5–7, 10, 15], were started to develop system architectures that enable comprehensive data acquisition and usage in the manufacturing environment. Since research emphasizes that especially small and medium-sized

J. Manner, S. Haarmann (Eds.): 12$^{\text{th}}$ ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

enterprises (SME) may increase their competitiveness through improvement of cooperation with federated companies (suppliers, purchasers, and subcontractors) [11], research projects started to consider an inter-organizational data exchange in their developed systems [5, 7, 10].

By analyzing the outcome of these researches, it is noticeable that all developed architectures show distributed system structures. Which means that parts of the systems are separated but work together like a single system.

Mishra & Tripathi [8] categorize distributed systems into systems where: a) only software and hardware are distributed, b) users are distributed and c) both – users, hardware and software are distributed. Since software development strategies changed from a fixed group of developers towards an open and community-based development, like in open source software projects, systems can also have distributed developers. Hence, the distributed system schema of Mishra & Tripathi can be enhanced by the category of distributed developers, as displayed in Figure 1.

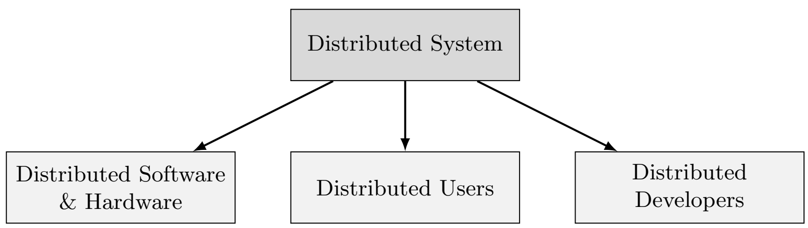


Fig. 1. Distributed Systems

Considering the schema that is displayed in Figure 1, a distributed system may belong to one up to three of the displayed categories. Each category implicates specific challenges that need to be considered [2, 3, 8].

Effective collaborations within distributed systems require adequate technical solutions. But especially for systems that have distributed users and/or developers, an adoption of good organizational practices and development processes is also necessary [3]. In this paper we do not focus on the technical and code-centric perspective of distributed system development, instead we are focusing on the social part of the development process and how to address it with technical solutions.

2 State of the Art

The integration of system users and developers with different knowledge in the development phase of a software is not a new aspect. All agile methods already implemented this aspect to enhance the field of knowledge and to get feedback early in the development phase [3]. In most agile software projects the participating developers are known. Therefore, it is easier to manage the cooperation between them. Software projects that are developed by distributed developers, e.g., via an open platform like GitHub, show the problem that system users and developers are unknown. This complicates the coordination and cooperation.

Allaho & Lee [1] analyzed software projects on GitHub in their research. They recognized that project documentation is very important to help developers to understand the project. It helps to create transparency of the project structure and behavior.

Transparency is a very important aspect, also for Carbot et al. [3]. They analyzed open source projects and occurring problems in their work and determined that most of the projects and its leadership miss transparency for the contributors. Furthermore, they recognized that most projects hardly ever follow any kind of democratic practices, which makes it difficult for contributors to influence the development of the project. During their research, Carbot et al. found out that many projects are only developed by a few members. It seems to be difficult to motivate others to contribute.

Promoting contribution not only refers to system developers, it also includes system users. Wikipedia is probably the most known example. There, a high amount of system users are motivated to contribute to the system [14]. Wikipedia provides a framework that makes it easy to contribute to the system even if the user is not a software developer. The principle of a wiki-based platform for communication and coordination of software projects was already implemented in different tools that support agile software development projects. It is a convenient method to exchange information between participants.

The importance of a medium for communication between system developers was also addressed in the work of Müller [9]. He pointed out that communication supports collaboration within a system's environment and motivates new contributors to get into the community.

3 Social Challenges in Distributed System Development

Based on the observations that were made, the social challenges in distributed software project can be summarized as:

C1 how to provide transparency (motivated by [1, 3, 14]),
C2 how to attract and support new contributors (mot. by [3, 9, 14]) and
C3 how to optimize collaborations (mot. by [3, 9]).

Transparency (C1) needs to be established in the entire distributed system. That means, architectural aspects (relating the hardware and software) need to be transparent as well as system contributions from users (e.g., feedback) or developers (e.g., system extensions). System transparency helps to understand the systems structure and behavior. Hence, it can be seen as the main social challenge in distributed systems.

System contributions (C2) are very important as they expand and improve the system. System transparency helps to minimize entry barriers for new contributors. Furthermore, the knowledge about a system's functional and operational range may inspire people to contribute new extensions and improvements.

The optimization of collaborations (C3) is important to maintain efficiency in system development and system operation. The system, or its environment, should provide a possibility (e.g., a tool) for users and developers to manage and optimize their collaborations.

4 Context and Assumptions

This section provides an overview of the context and the assumptions under which the development strategy, presented in the following section, was developed.

Since processes in the manufacturing industry are mostly triggered by occurring events (e.g., start and finish of tasks, error announcements, attainment of states, such as temperatures) it is assumed to have a system that has an event-driven architecture: information is published to a broker that routes it to subscribing services.

To support collaborations among small and medium-sized companies, an inter-organizational system usage should be supported by the system. That may enable companies to integrate machines and devices to the system and share information with a federated group of organizations.

Furthermore, the system should enable decentralized system development, where system components are developed by autonomous groups of software developers. They may develop new system components (services) that can be added to the system.

In order to develop a strategy that supports the development of such systems, in summary, we assume that:

- the system shows a distributed system structure in all categories (software & hardware, users, and developers)
- the system is based on an event-based approach (publish-subscribe pattern)
- the system is situated and developed in a trusted domain (no security considerations)
- the system will be developed in an agile manner (continuous system development and enhancement)

5 The Federated Development Strategy

To support the development of cross-organizational event-based systems, we developed a federated system development strategy that helps to handle the social challenges as they were stated in Section 3. To manage the challenges we developed an architecture of a Federated Management System (FMS) that needs to be integrated to the systems architecture.

Figure 2 displays an architectural sketch of our federated development strategy for cross-organizational event-based systems. In the upper part, the figure displays a system architecture that exchanges event messages between two companies via a broker (the assumed system situation). In this situation, social challenges appear because of the distributed system usage.

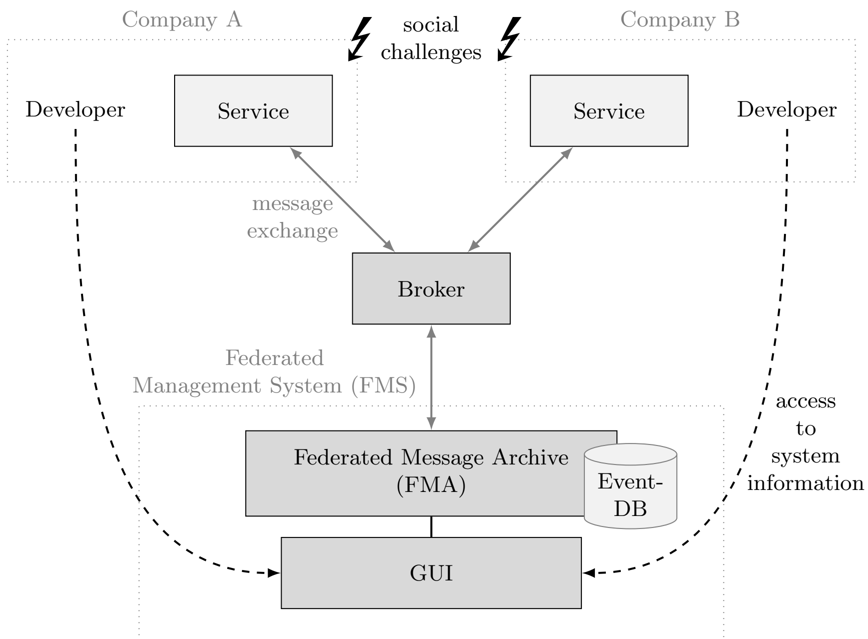


Fig. 2. Federated Management System

The lower part of the figure shows the FMS that needs to be integrated. It consists of a Federated Message Archive (FMA) component that includes an event-database and a graphical user interface (GUI). The FMA component is connected to the systems broker. Whereas other services only listen to specific events, the FMA subscribes to all messages that might be posted in the system. Consequently, the broker forwards all messages to the FMA which stores them in the event-database. In doing so, the FMA collects several information about posted events, such as message type, message type's publishing frequency, message content and more.

The GUI of the FMS, see Figure 3, provides access to the information in the FMA. It is the operative component that uncovers system information to users and enables transparency in the system environment. As displayed in Figure 3, a user may search for specific information that is shared via the system, e.g., information related to orders. Hence, the system displays a list of messages that are related to the search term. In this example the user can see that there exist more than one message type that is related to orders: topic '*order/new*' and

Fig. 3. Screenshot of an FMS GUI Prototype

topic '*order/accepted*'. Furthermore, the user can explore the messages content, e.g., the single attributes, in order to get the information he or she is locking for.

By searching for a specific message topic, the user can obtain further information about a specific message type. After forwarding a search query of a message topic to the FMA, all collected messages of the specific message type are compared and analyzed by the FMA component. Findings are formatted and sent to the GUI were they are displayed to the user. Hence, the user can obtain specific information about a message type, such as the frequency in which it is used or specific information about attributes that are held in the message content (e.g., range of attributes values or most common value).

The GUI of the FMA can be accessed by every developer that belongs to the trusted group of organizations in which the system is implemented (see Figure 1). This ensures a high degree of transparency within the whole system community (C1).

New developers that are interested in system contribution and join the cross-organizational system environment get access to the FMS. Due to transparency that is provided by the FMS, new system developers are able to get to know the systems structure and behavior. By analyzing events that were stored in the FMS, new developers can learn how messages are structured and how they can be subscribed in order to use their information for further purposes.

Besides the provision of system information, the FMS furthermore provides a wiki-based area where system developers are able exchange information. By this, system users get the opportunity to ask for special services and for system extensions within the system community. Concrete demands on system extensions may animate developers to contribute to the system (C2). Especially for new contributors it might be easier and more motivating to start contribution on a concrete task.

The wiki-based area of the FMS supports communication among the system participants. This is very important to optimize collaborations (C3). The wiki-based area may not only be used for placing demands, but also for coordinating contributions. Developers can quote on which functionality they are currently working and several developers are able to coordinate their collective work via, e.g., Kanban boards. Furthermore, new developers can ask for help within the community and senior system developers may share their knowledge and experiences.

6 Conclusion and Future Work

In this paper we presented an approach to handle social challenges in cross-organizational event-based systems. First, we presented a classification model of distributed systems which shows that distributed systems can be categorized into three categories: hardware & software distributed systems, as well as users distributed and developers distributed. Depending on their category of distribution, distributed systems face different challenges. Systems that have distributed users and/or developers have to handle not only technical and code-centric issues, but also social challenges. In this paper, we focused on social challenges and summarized them as follows: 1) provision of transparency, 2) attraction and support of new developers and 3) optimization of system collaborations. Finally, we presented FMS and how this system approach that can be used to mitigate social challenges in distributed systems.

In the future work it is planned to evaluate FMS in real system environments. To enable such evaluation, cross-organizational event-based systems will be implemented in the manufacturing environment and system developers will be asked to enhance the system by components that supports the manufacturing process. The evaluation of FMS's usability will be done in user studies that will be undertaken with the developers of the involved companies. There will be a mixture of developers with more and less experiences to establish the different needs and requirements to the FMS. Based on the evaluations outcome, FMS will be further adapted and improved.

References

1. Allaho, M.Y., Lee, W.C.: Trends and behavior of developers in open collaborative software projects. In: 2014 International Conference on Behavioral, Economic, and Socio-Cultural Computing (BESC2014). IEEE (Oct 2014)
2. Braubach, L., Pokahr, A.: Addressing Challenges of Distributed Systems Using Active Components. In: Intelligent Distributed Computing V, pp. 141–151. Springer Berlin Heidelberg (2011)

3. Cabot, J., Izquierdo, J.L.C., Cosentino, V.: Community-based software development for MDE tools. In: Joint Proceedings of EduSymp 2016 and OSS4MDE 2016 (2016)
4. Denkena, B., Dittrich, M.A., Uhlich, F., Maibaum, L., Mörke, T.: Das gentelligente Werkstück. In: Handbuch Industrie 4.0, pp. 295–321. Hanser Fachbuchverlag, Munich (2017)
5. DIN: Erweiterung des EPCIS-Ereignismodells um aggregierte Produktionsereignisse zur Verwendung in betrieblichen Informationssystemen (Jan 2016)
6. DIN: Reference Architecture Model Industrie 4.0 (Apr 2016)
7. Fuchs, J., Oks, S.J., Franke, J.: Platform-based service composition for manufacturing: A conceptualization. Procedia CIRP **81**, 541–546 (2019)
8. Mishra, K.S., Tripathi, A.K.: Some Issues, Challenges and Problems of Distributed Software System. In: (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (4) (2014)
9. Müller, M.: Agile Challenges and Chances for Open Source: Lessons Learned from Managing a FLOSS Project (Nov 2018)
10. Otto, B., Lohmann, S., Steinbuß, S., Teuscher, A.: IDS Reference Architecture Model Industrial Data Space. Tech. rep., International Data Space Association and Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V., Dortmund (2018)
11. Peris-Ortiz, M., Ferreira, J.J.: Cooperative and Networking Strategies in Small Business. Springer-Verlag GmbH (2016)
12. Reinhart, G., Zühlke, D.: Handbuch Industrie 4.0, chap. Von CIM zu Industrie 4.0, pp. XXXI–XL. Hanser Fachbuchverlag, Munich (2017)
13. Schuh, G., Reuter, C., Hauptvogel, A., Brambring, F., Hempel, T.: Ergebnisbericht des BMBF-Verbundprojektes PROSENSE : hochauflösende Produktionssteuerung auf Basis kybernetischer Unterstützungssysteme und intelligenter Sensorik, chap. 1 Einleitung, pp. 1–13. Apprimus Verlag Aachen, Aachen (2015)
14. van Steen, M., Pierre, G., Voulgaris, S.: Challenges in very large distributed systems. Journal of Internet Services and Applications **3**(1), 59–66 (Nov 2011)
15. Theorin, A., Bengtsson, K., Provost, J., Lieder, M., Johnsson, C., Lundholm, T., Lennartson, B.: An Event-Driven Manufacturing Information System Architecture for Industry 4.0. International Journal of Production Research **55**(5), 1297–1311 (Jul 2016)

BPMN in the Wild: BPMN on GitHub.com

Thomas S. Heinze[1], Viktor Stefanko[2], and Wolfram Amme[2]

[1] German Aerospace Center (DLR)
thomas.heinze@dlr.de
[2] Friedrich Schiller University Jena
[wolfram.amme,viktor.stefanko]@uni-jena.de

Abstract. We present our efforts in creating and analyzing a corpus of BPMN process models by mining software repositories. Systematically searching for BPMN artifacts in 6,163,217 repositories or 10% of all repositories hosted on GitHub.com, at the time of conducting our research, resulted in a diverse corpus of 8,904 BPMN 2.0 process models.

1 Introduction

Within the last years, an increasing number of software projects have shifted towards using platforms such as GitHub.com for their software development. Using these platforms as a source of data for empirical research allows for addressing a wide range of questions on the practice of software development and receives more and more attention, as indicated by the popularity of the flagship conference on the topic: *International Conference on Mining Software Repositories (MSR)*[1].

Research in the domain of business process modeling can as well benefit from such a data-driven approach. Due to characteristics of the domain, i.e., "process equals product", there is a lack of larger and commonly available datasets with real-world process models, which hinders empirical research in this area [5,6,8]. *Mining software repositories*, i.e., systematically retrieving, processing and analyzing process models from software repositories hosted on platforms such as GitHub.com, can help to overcome this lack. For example, research questions on how a language such as the *Business Process Model and Notation (BPMN)* [1] is used in practice can be addressed, in order to differentiate between the frequently and the rarely used parts of the language, thus advancing language and tool development. Analyzing modeling styles furthermore allows for investigating best practices and guidelines to help process designers. Eventually, best practices and tools as proposed by academic research or industry can be validated more realistically [7].

In this paper, we present our approach for mining software repositories on GitHub.com to create and analyze a corpus of BPMN process models. Due to the sheer number of repositories on GitHub.com and time constraints, we limited our approach to a randomly selected subset of 6,163,217 repositories or 10% of all software repositories on GitHub.com at the time of conducting the research. As a result, we were able to identify and analyze 8,904 distinct process models which are defined using BPMN 2.0's XML-based serialization format.

[1] http://www.msrconf.org

J. Manner, S. Haarmann (Eds.): 12th ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

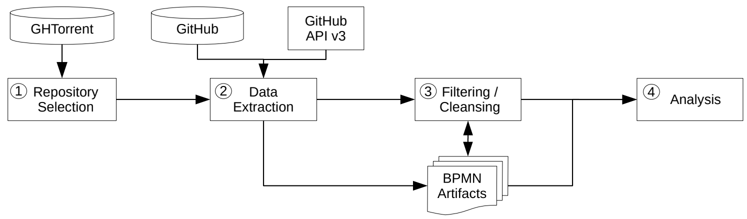


Fig. 1. Schematic illustration of the mining pipeline.

2 Related Work

The *Lindholmen dataset* has been an inspiration for this paper [2,7]. In Hebig et al. [2], the authors describe their approach to mine `GitHub.com` for UML models and report on gained insights. The dataset is considerably larger than our corpus, counting 93,596 models [7]. UML though is a family of general-purpose modeling languages while BPMN is one domain-specific modeling language. We are not aware of other work, which mines software repositories for BPMN models.

There have also been community efforts to create model collections [4]. The *BPM Academic Initiative* provides a platform to create and share business process models for academic teaching [5]. According to Ho-Quang et al. [4], the recent number of models is 29,285, but data collection has discontinued and the focus is on conceptual models as most models originate from students. Another initiative is the *BenchFlow project*, where business process models were collected from industrial partners. The authors claim to have collected 8,363 models, with a share of 64% of BPMN [8]. Unfortunately, the collection is not publicly available.

3 Mining BPMN on `GitHub.com`

Mining software repositories is a data mining task, consisting of steps of defining a research objective, selecting and extracting appropriate data, preprocessing and data cleansing, data analysis, and finally interpreting the analysis results.

In the first step of our implemented data mining pipeline, compare with Fig. 1, we got a list of all software repositories on `GitHub.com` by querying a local instance of the *GHTorrent*[2] database. We then randomly selected a subset of 6,163,217 non-forked repositories. All 6,163,217 repositories were examined for potential BPMN artifacts using the *GitHub API*[3] in the second step. To this end, the default branch and its file structure were queried for each repository. Potential BPMN artifacts were then identified based on their file name and file extension. Among the analyzed repositories, we found 1,251 repositories, with at least one potential BPMN artifact and overall 21,306 artifacts. We downloaded the identified repositories and artifacts. In the third step, since the artifacts

[2] `http://ghtorrent.org/`
[3] `https://developer.github.com/v3`

included a wide range of file formats, we filtered for BPMN 2.0's XML-based serialization format, which lowered the number of artifacts to 16,907. Additionally removing duplicates, yielded the corpus of 8,904 distinct BPMN 2.0 models. All the BPMN artifacts were finally subject to a preliminary analysis in the fourth step. Information on the corpus and analysis outcomes are available online [3][4].

4 Preliminary Analysis

In our preliminary analysis, we were mainly interested in the diversity of the found BPMN artifacts. We here sketch some of the results. Looking at the artifacts' age, more than each third was modified in the last year at the time of conducting our research. We though also found artifacts older than 8 years. Using the locations of contributors allowed us to reason on the artifacts' geographical origin, where China, USA, and Germany played prominent roles. The corpus spans a range of different model sizes. While half of the processes are smaller than 20 nodes, we identified 57 models with more than 1,000 nodes. We were also able to confirm the finding in [2], that models play a rather static role in software repositories. Up to three quarter of all the BPMN artifacts were never updated at all.

We think that our corpus of BPMN models provides a starting point for understanding more about the practice of BPMN. In future work, besides increasing the coverage of analyzed repositories, we want to research on questions about BPMN's use on `GitHub.com`, e.g., what are frequently and rarely used constructs or are there certain characteristics that can be used to predict modeling errors [5].

References

1. Business Process Model and Notation (BPMN), Version 2.0. Object Management Group (OMG) Standard (2011), https://www.omg.org/spec/BPMN/2.0/PDF
2. Hebig, R., Quang, T.H., Chaudron, M., Robles, G., Fernandez, M.A.: The Quest for Open Source Projects that use UML: Mining GitHub. In: MODELS 2016. pp. 173–183. ACM (2016)
3. Heinze, T.S., Stefanko, V., Amme, W.: Mining von BPMN-Prozessartefakten auf GitHub. In: KPS 2019. pp. 111–120. DHBW Stuttgart (2019)
4. Ho-Quang, T., Chaudron, M.R.V., Robles, G., Herwanto, G.B.: Towards an Infrastructure for Empirical Research into Software Architecture: Challenges and Directions. In: ECASE@ICSE 2019. pp. 34–41. IEEE (2019)
5. Kunze, M., Luebbe, A., Weidlich, M., Weske, M.: Towards Understanding Process Modeling – The Case of the BPM Academic Initiative. In: BPMN 2011 Workshops. pp. 44–58. Springer (2011)
6. Lübke, D., Pautasso, C.: Empirical Research in Executable Process Models. In: Empirical Studies on the Development of Executable Business Processes, pp. 3–12. Springer (2019)
7. Robles, G., Ho-Quang, T., Hebig, R., Chaudron, M., Fernandez, M.A.: An extensive dataset of UML models in GitHub. In: MSR 2017. pp. 519–522. IEEE (2017)
8. Skouradaki, M., Roller, D., Leymann, F., Ferme, V., Pautasso, C.: On the Road to Benchmarking BPMN Workflow Engines. In: ICPE 2015. pp. 301–304. ACM (2015)

[4] https://github.com/ViktorStefanko/BPMN_Crawler

Towards a Framework for Context-Aware Resource Behaviour Analysis

Maximilian Völker[1] and Luise Pufahl[2]

[1] Hasso Plattner Institute, University of Potsdam, 14482 Potsdam, Germany
maximilian.voelker@student.hpi.de
[2] Software and Business Engineering, Technische Universitaet Berlin, Berlin, Germany
luise.pufahl@tu-berlin.de

Abstract. For the successful and efficient execution of business processes, resources are essential. However, it is difficult to predict or plan executions appropriately, as the behaviour of resources, especially human workers, highly varies depending on the individual and the context. Although there are several metrics to describe resource behaviour in research, the reasons for their behaviour and the influence of the environment, like the workload, have been less explored. Extracting resource-related metrics from event logs and analysing them for possible relationships opens the opportunity to understand resource behaviour and improve working conditions. In this work, a framework for analysing correlations between resource behaviour and environment is motivated and briefly sketched.

Keywords: Resource Behaviour, Business Processes, Process Mining

1 Introduction

Resources play a crucial role for the correct execution of business processes [2] and their behaviour heavily affects the overall performance of the processes they are involved in [4]. But unlike machines, human resources do not show constant behaviour at work: their working speed varies, they might batch work or are only available part-time [9]. In addition, humans have different preferences regarding their work-items or co-workers, which is reflected in their behaviour [1].

From a temporal perspective, workers most likely change their behaviour and preferences over time due to personal development or adjustments to a new environment or circumstances. In the area of work psychology, for example, the arousal, i.e. stress, of workers is recognised to be related to their performance, known as the Yerkes-Dodson law [10].

In the context of business process technology, the behaviour and decisions of resources, as well as process-related circumstances, are incidentally captured in event logs. Metrics like workload, processing speed, waiting times and preferences in terms of task selection can, for example, be derived from the event log [1, 6], provided resource information is available for tasks. However, even though many researchers state that human resources and their behaviour greatly affect overall

J. Manner, S. Haarmann (Eds.): 12$^{\text{th}}$ ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

process performance, there is only little research on mining and, more importantly, understanding the behaviour of human resources in the context of the process execution [7].

In the remainder of the paper, related work regarding resource metrics is presented in Sect. 2. Section 3 introduces the concept for a new framework for resource behaviour analysis.

2 Foundations

So far, several metrics to measure the behaviour and performance of human resources have been proposed. For example, Swennen et al. [8] introduce the notions of *Resource Frequency*, *Resource Involvement*, and *Resource Specialisation*, indicating how active resources are and in how many cases they participate. In terms of resource behaviour, Suriadi et al. [7] describe how to extract the queuing discipline (bounded to FIFO, LIFO or Priority) that resources show and Martin et al. [3] propose an approach for detecting batching in resource behaviour. In addition, Pika et al. [6] provide examples for metrics in the following categories: *Skills*, *Utilisation*, *Preference*, *Productivity* and *Collaboration*.

Although several papers describe different metrics for resource behaviour, only a few consider them in context. But former research already showed that correlations between resource-related metrics can be found in process logs: Nakatumba et al. [4] confirmed the Yerkes-Dodson law by extracting the workload and processing times from process logs and performing a regression analysis.

Another exception in the context of correlating resource metrics is the comprehensive framework developed by Pika et al. [6]: They present an approach for extracting time series of Resource Behaviour Indicators (RBI) from event logs using SQL-like queries. In later work, this framework was extended by the aspect of the "relationship between resource behaviors and outcomes"[5] by including a regression analysis of their RBIs. However, there are some points for improvements, e.g. regarding the scope and complexity of the metrics available for analysis and the reuse or export of calculations.

3 A Framework for Context-Aware Resource Behaviour Analysis

Due to the limitations of existing work, we plan to develop a framework for context-aware resource analysis with a three-step approach as shown in Fig. 1.

Metric Selection For examining the behaviour of resources not only the analysis-part plays an important role but also the metrics themselves must be considered in detail beforehand. Metrics are measurements used to quantify performance aspects and can be calculated from data for a point in time or time spans. In the context of resources and processes, examples are the number of activities a resource is working on, or how many activities are assigned to a resource but have not yet been started.

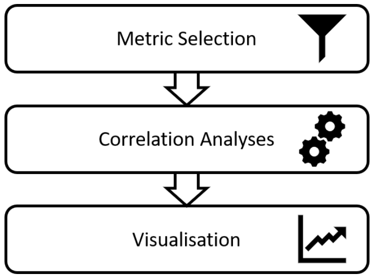


Fig. 1. Framework Steps

The framework will include, but not be limited to, a collection of resource-related metrics from the literature. To guide the selection of metrics for analysis, we will furthermore classify them into *environmental* metrics (influencing behaviour) and *behavioural* metrics (expressing behaviour), which should support more targeted and meaningful analyses. Additionally, the framework will not be limited to directly resource-related metrics, since case-related or event-related information, such as the case duration or the time of day, may also have an effect on the behaviour or decisions of resources and will therefore be available for analysis as well.

Each metric comes with its own extraction logic and imposes, often implicitly, certain requirements on the data set, such as certain attributes or meta-information needed for computation. However, requirements for the process log are often not mentioned in literature. Besides these demands, metrics can also have different calculation techniques that differ in their requirements and quality based on assumptions. The processing time, for example, could be extracted by taking the timestamps of start and end events into account, or the required time is specified directly in the log as an attribute. Some logs may even lack this information, but an estimation of processing times could still be made, e.g. by considering the subsequent event and assuming waiting times. The framework for computing such metrics should therefore be aware of these variations and prerequisites and be extendable with new metrics and calculation techniques. This allows for a flexible and general application on a wide range of event log variants.

Correlation Analyses After the metric-computation, correlation analysis can be used to determine if there is a relationship between them. By automatically executing the analyses for selected metrics, the framework is able to reveal interesting insights for further manual investigation. For this, the separation into environmental metrics and behavioural metrics might help to detect more relevant results, as it indicates the direction of possible causalities.

To enable future research based on resource behaviour, the data and time series calculated by the framework should be exportable, e.g. by enriching the process log with new data and attributes, such as the workload or the current work prioritisation pattern. This would facilitate further processing of the data series, e.g. with techniques from the field of machine learning. The resulting models could be used not only to anticipate the resources' reactions to impending environmental changes but also to achieve a more powerful and realistic process simulation regarding resources.

Visualisation The visualisation component plays an important role as it is used to communicate the outcome of the analysis. On the one hand, it should include the resulting numbers and graphs for a comprehensive evaluation by experts; on the other hand, the visualisation should quickly point out interesting findings and provide assistance in interpreting the results.

The concept for a new framework for analysing resource behaviour based on event logs as presented in this paper suggests and encourages further research on this topic. There are several points for future work, including a comprehensive and practical overview of resource-related metrics or new possibilities to combine and analyse metrics, also with regard to other research areas, such as psychology.

References

1. Bidar, R., ter Hofstede, A.H.M., Sindhgatta, R., Ouyang, C.: Preference-based resource and task allocation in business process automation. In: Panetto, H., Debruyne, C., Hepp, M., Lewis, D., Ardagna, C.A., Meersman, R. (eds.) On the Move to Meaningful Internet Systems: OTM 2019 Conferences, Lecture Notes in Computer Science, vol. 11877, pp. 404–421. Springer International Publishing, Cham (2019)
2. Cabanillas, C.: Process- and resource-aware information systems. In: Matthes, F., Mendling, J., Rinderle-Ma, S. (eds.) 2016 IEEE 20th International Enterprise Distributed Object Computing Conference (EDOC). pp. 1–10. IEEE, Piscataway, NJ (2016)
3. Martin, N., Swennen, M., Depaire, B., Jans, M., Caris, A., Vanhoof, K.: Retrieving batch organisation of work insights from event logs. Decision Support Systems 100, 119–128 (2017)
4. Nakatumba, J., van der Aalst, W.M.P.: Analyzing resource behavior using process mining. In: Rinderle-Ma, S. (ed.) Business Process Management Workshops, Lecture Notes in Business Information Processing, vol. 43, pp. 69–80. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg (2010)
5. Pika, A., Leyer, M., Wynn, M.T., Fidge, C.J., ter Hofstede, A.H.M., van der Aalst, W.M.P.: Mining resource profiles from event logs. ACM Transactions on Management Information Systems (TMIS) 8(1), 1 (2017)
6. Pika, A., Wynn, M.T., Fidge, C.J., ter Hofstede, A.H.M., Leyer, M., van der Aalst, W.M.P.: An extensible framework for analysing resource behaviour using event logs. In: Hutchison, D., Kanade, T., Kittler, J., Haralambos, M., Horkoff, J. (eds.) Advanced Information Systems Engineering, Lecture Notes in Computer Science / Information Systems and Applications, Incl. Internet/Web, and HCI, vol. 8484, pp. 564–579. Springer International Publishing, Cham (2014)
7. Suriadi, S., Wynn, M.T., Xu, J., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Discovering work prioritisation patterns from event logs. Decision Support Systems 100, 77–92 (2017)
8. Swennen, M., Martin, N., Janssenswillen, G., Jans, M.J., Depaire, B., Caris, A., Vanhoof, K.: Capturing resource behaviour from event logs. In: Ceravolo, P., Guetl, C., Rinderle-Ma, S. (eds.) Proceedings of the 6th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2016). pp. 130–134. CEUR Workshop Proceedings, RWTH Aachen, Aachen, Germany (2016)

9. van der Aalst, W.M.P., Nakatumba, J., Rozinat, A., Russell, N.: Business process simulation. In: vom Brocke, J., Rosemann, M. (eds.) Handbook on Business Process Management 1, pp. 313–338. Handbook on business process management, Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
10. Wickens, C.D., Hollands, J.G.: Engineering psychology and human performance. Prentice-Hall, Upper Saddle River, NJ, 3. ed. edn. (2000)

Towards the Discovery of Object-Aware Processes

Marius Breitmayer and Manfred Reichert

Institute of Databases and Information Systems, Ulm University, Germany
{marius.breitmayer,manfred.reichert}@uni-ulm.de

Abstract. There has been a huge body of research in order to reduce manual efforts in creating executable process models through the automated discovery of process models from the event logs created by information systems. Regarding activity-centric processes, such event logs comprise case ids and events related to the execution of process activities. However, there exist alternative process management paradigms, such as object-aware processes, for which existing algorithms fail to discover a sound model. These algorithms do not treat data as first-class citizens, but solely rely on the information from event logs. In consequence, existing discovery algorithms are insufficient for discovering object-aware processes. To address this issue, discovery algorithms need to consider additional data sources (e.g., existing forms). This paper discusses the need for dedicated discovery techniques in object-aware processes.

Keywords: object-aware processes, process mining, process discovery

1 Introduction

Despite the many mining approaches that exist for activity-centric processes, adequate support for discovering data-centric process models, e.g., in the context of artifact-centric processes [7], case handling [2], or object-aware processes [8], is still lacking. While an activity-centric process model consists of a sequence of activities that need to be executed in a defined order, a data-driven and -centric process allows for greater flexibility through the use of declarative process rules and generated forms [15,6]. Current process discovery algorithms are able to discover the schema of an activity-centric process from an event log, whereas information about the internal logic of the activities (e.g., user forms or data required for an activity) is often neglected. As data is treated as first-class citizen in data-centric (e.g., object-aware) process management, the discovery of corresponding models should consider this issue as well.

To understand the nature of the problem at hand, a short introduction into data-centric and object-aware process management becomes necessary. PHILharmonicFlows, our approach to data-centric processes, introduces the concepts of *objects*, *object behavior,* and *object interaction.* For each business object present in a real-world business process, one such object exists. The latter comprises data, represented by *attributes,* and a state-based process model describing the

J. Manner, S. Haarmann (Eds.): 12^{th} ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

object behavior in terms of an *object lifecycle* model. When data becomes available during runtime, this enables transitions between the various states of the lifecycle process, i.e., execution is data-driven. In the e-learning system *PHoodle,* a practical application of the PHILharmonicFlows system [5], examples of business objects include *Submission*, *Exercise*, and *Lecture* (see Fig. 1 for the respective data model). In turn, when the values of certain attributes, such as *Points* or *Feedback,* become available at runtime, this enables the transition between the states of a lifecycle process (see Fig. 2). Finally, the interactions between object lifecycles are managed by coordination processes [14].

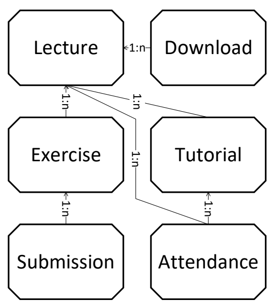


Fig. 1. Data Model

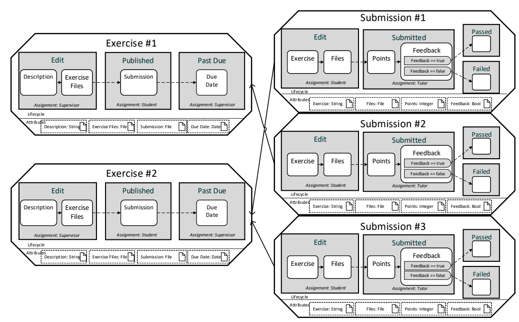


Fig. 2. Objects with Lifecycles and Interaction

2 Related Work

Process discovery summarizes techniques that leverage information from event logs to discover process models [3]. For activity-centric processes, there exist a variety of approaches (see [1,17] for an overview). Various algorithms use event logs as input to discover an activity-centric process model. Regarding data-centric processes [16], however, there only exist few approaches for discovering process models. [12] describes an approach for discovering artifacts and their lifecycles from structured datasets as opposed to lifecycle-enabled objects in our approach. In turn, [9] deals with methods for discovery of artifacts and the interactions between them; additionally, an evaluation based on real-life datasets from ERP systems is provided. In turn, [13] decomposes the problem of artifact lifecycle discovery such that existing process mining algorithms can be applied. The construction of data and object models from different data structures (e.g., databases, legacy systems) has been investigated in reverse engineering [4,10]. While database reverse engineering reconstructs logical or conceptual models, other aspects of data-driven process management are neglected (e.g., lifecycles or the interactions between object lifecycles). An approach to automatically generate event logs from databases is described in [11]. Since data is treated as a first-class citizen in object-oriented process management, additional information (i.e., data sources) need to be considered to discover an object-aware process.

3 Research direction

In our PHILharmonicFlows framework, an object-aware process consists of a *data model*, one *lifecycle model* for each object, and a *coordination process* enforcing constraints regarding object interactions [8]. In order to discover an executable object-aware process, all three aspects need to be considered. For the discovery of various aspects of object-aware processes (e.g., relations between objects or states of a lifecycle), solely considering event logs is not sufficient and, hence, additional data sources need to be taken into account. For example, the data model underlying an object-aware process provides the foundation for both lifecycles and object coordination [5].

The **first step** during process discovery is to identify objects, including their attributes and relations. Note that the structure of a normalized relational database, to a certain degree, is comparable to a data model, which offers the opportunity to discover the data model from the structure (i.e., the create table statements) of a database. Each table in the database may, but does not have to correspond to an object in the data model, whereas columns of a table represent the attributes of an object. One-to-many relations between tables can be used to identify relations between the objects of a data model. Additionally, relations can be used as an indicator if a table corresponds to a correct object.

After discovering the data model, the object lifecycles need to be discovered in the **second step**. Based on the attributes from the data model, lifecycle discovery shall deliver object states as well as the transitions between them. In general, a lifecycle process may enter another state, if all necessary data (i.e., attribute values) are available. In particular, lifecycle states cannot be discovered from event logs, whose entries solely refer to activities due to the mismatch between states (i.e., defined by attributes) and cases (i.e., a collection of activities). To tackle this mismatch, discovery algorithms for object lifecycles, suitable event log preprocessing (e.g., splitting an event log into event logs for each object), and additional data sources (e.g., forms of existing information systems) need to be considered as well during the discovery process.

The **third step** in discovering an object-aware process is to unravel the coordination of interactions between objects (e.g., a submission may only be created if the corresponding exercise is in state published). As object interaction can only be discovered with the data model and the lifecycles present, their discovery is a secondary problem for now.

4 Conclusion

This paper discusses the need for spending research efforts on the discovery of object-aware process models. As major advantage, the discovery of an object-aware processes allows to identify the underlying logic of a process. Finally, due to the strong linkage between process and data in object-aware processes, it is possible that not every aspect of each element (i.e., data model, lifecycles, and coordination) may be discovered from the presented data sources and, therefore further research is of utmost importance.

References

1. van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer, 2 edn. (2016)
2. van der Aalst, W.M.P., Weske, M., Grünbauer, D.: Case handling: a new paradigm for business process support. DKE **53**(2), 129–162 (2005)
3. van der Aalst, W.M.P., et al.: Process mining manifesto. In: Int'l Conf on BPM'11. pp. 169–194 (2011)
4. Alhajj, R.: Extracting the extended entity-relationship model from a legacy relational database. Information Systems **28**(6), 597 – 618 (2003)
5. Andrews, K., Steinau, S., Reichert, M.: Engineering a highly scalable object-aware process management engine using distributed microservices. In: Int'l Conf on CoopIS'18. pp. 80–97 (2018)
6. Andrews, K., Steinau, S., Reichert, M.: Enabling runtime flexibility in data-centric and data-driven process execution engines. Information Systems p. 101447 (2019)
7. Cohn, D., Hull, R.: Business artifacts: A data-centric approach to modeling business operations and processes. IEEE Data Eng. Bull. **32**(3), 3–9 (2009)
8. Künzle, V., Reichert, M.: PHILharmonicFlows: towards a framework for object-aware process management. J of Soft Maint & Evo **23**(4), 205–244 (2011)
9. Lu, X., Nagelkerke, M., van de Wiel, D., Fahland, D.: Discovering interacting artifacts from ERP systems. IEEE Trans Serv Com **8**(6), 861–873 (2015)
10. Mfourga, N.: Extracting entity-relationship schemas from relational databases: a form-driven approach. In: WCRE'97. pp. 184–193 (1997)
11. de Murillas, et al.: Case notion discovery and recommendation: automated event log building on databases. Know & Inf Sys (2019)
12. Nooijen, E.H.J., van Dongen, B.F., Fahland, D.: Automatic discovery of data-centric and artifact-centric processes. In: Int'l Conf on BPM'12. pp. 316–327 (2012)
13. Popova, V., Fahland, D., Dumas, M.: Artifact lifecycle discovery. Int'l J of Coop Inf Sys **24**(01), 1550001 (2013)
14. Steinau, S., Andrews, K., Reichert, M.: Modeling process interactions with coordination processes. In: CoopIS'18. pp. 21–39. LNCS, Springer (2018)
15. Steinau, S., Andrews, K., Reichert, M.: Executing lifecycle processes in object-aware process management. In: Data-Driven Process Discovery and Analysis. pp. 25–44. Springer (2019)
16. Steinau, S., Marrella, A., Andrews, K., Leotta, F., Mecella, M., Reichert, M.: DALEC: A framework for the systematic evaluation of data-centric approaches to process management software. Softw & Sys Modeling **18**(4), 2679–2716 (2019)
17. Weerdt, J.D., Backer, M.D., Vanthienen, J., Baesens, B.: A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. Inf Sys **37**(7), 654 – 676 (2012)

When to use FaaS? - Influencing technical factors for and against using serverless functions

Robin Lichtenthäler, Stefan Winzinger, Johannes Manner, and Guido Wirtz

Distributed Systems Group, University of Bamberg
robin.lichtenthaeler@uni-bamberg.de
stefan.winzinger@uni-bamberg.de
johannes.manner@uni-bamberg.de
guido.wirtz@uni-bamberg.de

Abstract. Cloud computing offerings evolve continuously. A recent trend is the Function as a Service paradigm which confronts developers with the decision whether adopting this new paradigm can be beneficial for parts of their application. However, many factors influence this decision or even prevent the usage of FaaS. Therefore, this paper provides a structured overview of relevant technical factors to guide the decision process.

Keywords: FaaS, serverless, serverless function, cloud computing, decision

1 Introduction

Building applications which run in a cloud environment is the norm for many enterprises today [9]. However, the possibilities how applications are designed and run in the cloud are manifold and evolve with new offerings of cloud providers. A recent trend is serverless computing with the Function as a Service (FaaS) offering at its core [4]. Developers implement a *serverless function* by writing code, potentially combined with further code artifacts (e.g., libraries), which processes an input and produces an output according to a predefined interface. The serverless function can then be deployed to a *FaaS platform* which enables its execution. The FaaS platform stores the function code and registers it to be executable via one or more triggers. The available triggers depend on the specific platform [17]. Typical triggers are messages from a messaging system, database events or an HTTP request to a URL associated with the function by the FaaS platform. When a serverless function is invoked via such a trigger, the platform manages the actual execution transparently. That means, the platform starts up the execution environment with the required resources, typically a container, and executes the function with the given input [4]. For multiple invocations, running containers can be reused or the platform can scale accordingly by starting or stopping additional containers. Whether a platform can reuse a container (warm start) or has to start up a new container (cold start) can have a significant impact on the execution time [19].

J. Manner, S. Haarmann (Eds.): 12th ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

FaaS platforms are typically administered by cloud providers as in the case of AWS Lambda[1] or Azure Functions[2]. These cloud providers also offer direct integration with their respective cloud services, like database services or messaging services to serve as triggers or dependencies for functions. But also open-source platforms like OpenFaaS[3] or Knative[4] are available.

FaaS enables a fine-granular billing in a commercial setting where customers only pay when a serverless function is actually invoked and executed [2, 10]. Since the FaaS platform starts and stops function containers transparently in order to scale the system dynamically and to enable this billing model, serverless functions should be stateless. This statelessness is one technical aspect in the decision for or against using serverless functions. Although such aspects are already known, they were not summarized yet and are often discussed in a different context [4, 10, 15]. Therefore, the aim of this paper is to provide a structured and comprehensive overview of criteria relevant for the decision process. The focus is on criteria which result from the technical characteristics of FaaS platforms. Since it can make sense to use serverless functions only for parts of an application, the starting point for the criteria we consider is a so-called functionality.

A *functionality* represents a specific part of business logic which can be implemented and potentially deployed as a serverless function. An *application* is a composition of multiple functionalities. An application primarily consisting of serverless functions is called a *serverless application*.

In the following, section 2 provides a short overview of our approach for the identification of the criteria which are presented in a catalog as the main contribution of this paper in section 3. We discuss characteristics of the criteria and how they can be practically applied in section 4. Section 5 gives an overview of related work whereas we conclude the paper in section 6.

2 Approach

The initial identification of relevant technical criteria was not based on a formal approach but our prior experience with studying FaaS as a recent trend in cloud computing. In order to refine and validate the identified criteria, we then decided to build upon the work of Spillner et al. [23] who maintain a comprehensive list of research on serverless computing. By relying on their collection of relevant literature, we looked for insights that either substantiated or rebutted our identified criteria. The result is the criteria catalog in section 3. The basic assumption is that each criterion can be evaluated on the basis of a functionality which is considered to be implemented as a serverless function.

[1] https://aws.amazon.com/lambda
[2] https://azure.microsoft.com/en-gb/services/functions/
[3] https://www.openfaas.com/
[4] https://knative.dev/

3 Criteria catalog

Statelessness: A functionality should be stateless since the FaaS platform cannot guarantee a reexecution on the same instance [15, 21]. This means that a functionality should not hold state (e.g., data stored in memory or on the disk of the function instance) of previous invocations which is required for further invocations. Thus, a serverless function is an example of the *stateless component* pattern [12] and all state required should be externalized.

Idempotency: A functionality should be idempotent in the sense of the *idempotent processor* pattern [12] meaning that it can be reexecuted multiple times with the same input and produce the same output. This mostly applies to how a serverless function handles externalized state. Since FaaS platforms typically reexecute a function in case of an error, a functionality should be idempotent or at least provide the possibility to be made idempotent [15]. Otherwise, a reexecution can lead to unexpected behavior, for example data inconsistencies because of repeated database transactions.

Synchronous dependencies: Synchronous dependencies express the frequency a functionality requires synchronous communication with other services during its execution. This means that a request is sent to an external system, e.g., a data storage or an external API, and the process is blocked until a response arrives. Because a serverless function is also charged for the time in which it is blocked, each synchronous dependency potentially creates unnecessary cost [3, 5]. This effect can even be increased when multiple function instances are executed on the same host and therefore also have to share the available network bandwidth for their synchronous dependencies [13]. A functionality having many synchronous dependencies should therefore not be implemented as a serverless function. Asynchronous interactions with other services where no response is expected are not problematic.

Event-driven architecture: An event-driven architecture aligns well with the FaaS paradigm because serverless functions are well-suited as event processors [5, 10, 20]. Therefore, if an event-driven architecture is used for the application the functionality belongs to, a serverless function is a suitable implementation. Additionally, several FaaS platforms provide a built-in support (in the form of specific triggers) for the consumption of events from various sources.

Algorithmic computing resource efficiency: The computing resources required for the execution of a functionality depend on its algorithmic characteristics and the input. Since the computing resources have to be defined upfront for serverless functions, a computing resource configuration has to be chosen which is able to handle the input requiring the most computing resources. Consequently, potential inputs need to be known or a worst-case assumption for the input has to be made. Memory and CPU power are typically not assigned independently but change in conjunction [18] for most FaaS platforms. Therefore, the configuration has to be chosen so that enough CPU power and memory are assigned meaning that usually one of the two is oversupplied for most inputs.
Oversupplied CPU power does not inevitably result in higher cost because less complex inputs are often processed faster [16]. If additional cores are assigned

with more CPU power, it depends on the ability of the functionalities to be processed by several cores in parallel. However, oversupplied memory does not result in a faster execution. It leads to unnecessary cost if memory is provisioned and has to be paid for, although not needed for all inputs [7, 8]. Thus, an inefficient usage of the memory resources contradicts the implementation as a serverless function from an economical point of view.

Maximum execution time: The execution time for a serverless function has an upper limit enforced by the FaaS platform [4, 17] because serverless functions should be short-lived [10, 13]. The maximum execution time for a functionality, even with the highest computing resource configuration, has to be lower than this limit. Otherwise, the execution will abort with a timeout.

Maximum memory consumption: FaaS platforms limit the memory which can be assigned to the execution of a function [4, 17]. The maximum memory consumption of a function for all inputs has to be smaller than the maximum possible memory configuration. Otherwise, the execution will abort with an out of memory error.

Availability of execution environments: Available execution environments are managed by the FaaS platform [15]. Implementing a functionality as a serverless function is therefore only possible if the desired execution environment is supported by the chosen FaaS platform. While the available execution environments of AWS Lambda are limited to common execution environments like Java, Python or NodeJS [17], OpenFaaS accepts custom Docker images enabling the usage of arbitrary environments [22].

Deployment artifact size: FaaS platforms limit the maximum size of the deployment artifact (code archive, container image) for a serverless function [17] to prevent too large functions and excessive storage usage. If the size of a serverless function implementation in the form of its deployment artifact exceeds this limit, it is not possible to implement the functionality as a serverless function.

Latency: If a functionality strictly requires a very low latency, a FaaS platform might not be able to provide such a latency because of the start-up overhead for starting a function instance. This start-up overhead is the time needed by the FaaS platform to provide the required computing resources and to start the infrastructure (e.g., the container). Since the FaaS platform manages transparently when function instances are started and stopped, the start-up overhead occurs frequently [21]. In general, this leads to a longer response time for functionalities implemented as serverless functions [1–3]. Although the overhead is also influenced by the runtime environment, it is inherent to the FaaS paradigm. Therefore, the suitability of serverless functions depends on the latency requirements for a functionality, as discussed in detail by Aditya et al. [1].

Update frequency: Running systems have to be updated frequently, e.g., to implement new features or to fix bugs. Using FaaS offers benefits regarding the speed with which functionalities can be updated. These benefits are based on two specific aspects of FaaS: externalization of operational tasks and the small size of independent deployment units. Because tasks like hardware installation, operating system maintenance and container orchestration are externalized

[10, 11], no additional efforts are required and developers can focus solely on the actual functionality [15]. All serverless functions of an application can be deployed independently. If a functionality has to be changed, only the serverless function implementing this functionality has to be redeployed which saves time in contrast to other approaches.

Vendor independence: In order to be vendor-independent, the possibility to implement and deploy a functionality in different environments is needed. FaaS platforms expect that serverless functions are implemented according to an interface predefined by the corresponding platform provider. Once a functionality is implemented as a serverless function for a specific FaaS platform, its code has to be adapted if it has to be transferred to another FaaS platform [2]. Furthermore, FaaS platforms provide platform-specific services which are often used by the functions. If these services are used, additional efforts have to be made to transfer and provide these services on another platform [15, 24]

Workload type: In general, different workload types can be distinguished which can be used to classify specific usage profiles of functionalities. According to Fehling [12], there are static, periodic, once-in-a-lifetime, unpredictable and continuously changing workloads. If the workload is unpredictable, i.e., bursty, or continuously changing, FaaS is well-suited since its provision of resources can be adapted dynamically [4]. Under- or over-provisioning does not occur compared to other deployment options like VMs. For a once-in-a-lifetime workload (e.g., for a migration) FaaS makes sense if the resources required cannot be provided otherwise, but it is not the typical use case for FaaS. Handling a static workload can usually be implemented more cost-efficiently in a more traditional cloud model since the resources needed are known upfront and are used most of the time which is cost-efficient [15]. If workload is produced periodically, FaaS can make sense but offers no specific advantage compared to other deployment models. Apart from this broad assessment, a thorough cost calculation for a specific usage profile is required to definitively evaluate the suitability of FaaS [7].

4 Discussion

The criteria presented in section 3 cover important aspects of FaaS in a compact way without being too detailed. Therefore, other criteria were excluded from our collection which are not exclusive to FaaS. E.g., for a cloud service like AWS Lambda, it might not be possible to determine where the serverless function is actually executed. Howevere, there can be regulations for sensitive date that they can only be processed in certain countries. Thus, the usage of FaaS for sensitive data might be prohibited. Despite being an important concern, it is not exclusive to FaaS because it also applies to other cloud deployment models like Platform as a Service (PaaS). Another often mentioned advantage of FaaS is the automated horizontal scalability of serverless functions. While it can be a convincing argument for FaaS, it is not an exclusive characteristic of FaaS. Other deployment models like a Kubernetes[5] deployment with a horizontal pod

[5] https://kubernetes.io/

autoscaler or a PaaS deployment can offer the same horizontal scalability, although lacking the possibility to scale to zero instances. Additional criteria in this regard are availability of SLAs and resilience. It has to be noted that the decision for or against using FaaS therefore also depends on the alternatives to which FaaS is compared to. Alternative deployment models could for example be an on-premises deployment on dedicated hardware, a virtual machine (VM) deployment within an Infrastructure as a Service (IaaS) offering, a PaaS deployment, or a deployment in a Kubernetes cluster. For some of those alternatives the mentioned excluded criteria can make a significant difference. But, as already stated, to keep the catalog comprehensible we decided to focus on criteria exclusive to FaaS and a more comprehensive comparison of criteria with selected deployment alternatives could be done in future work.

Furthermore, our catalog is focused on criteria resulting from the technical characteristics of FaaS. Other non-technical criteria can be relevant in a decision process as well, for example the effort to train developers to work with the new paradigm and the specific FaaS platforms. But criteria which cannot be regarded on the basis of a single functionality are out of the scope for this work because they are less helpful for the decision whether FaaS is applicable to parts of an application.

An additional aspect regarding the criteria included in our catalog is that their relevance differs depending on whether an administered FaaS platform of a cloud provider or a self-hosted open source platform is used. This should be kept in mind when applying the criteria in a decision process since operational concerns are not completely externalized when a self-hosted platform is used.

Table 1. Decision guidance for incorporating serverless functions

Category	**Criterion**	**Dimension**
Mandatory	Statelessness	Yes/No
	Executable in maximum execution time	Yes/No
	Executable with desired memory setting	Yes/No
	Availability of desired execution environment	Yes/No
	Deployment artifact size not exceeded	Yes/No
Cost-Efficiency	Synchronous dependencies	Number of dependencies
	Algorithmic computing resource efficiency	% of Utilization
	Expected usage profile	static/periodic/once-in-a-lifetime/unpredictable
Individual	Effort to achieve idempotency	high/medium/low
	Event driven architecture	Yes/No
	Update frequency	Number of deployments per month
	Vendor independence important	Yes/No

We have summarized the criteria as a structured questionnaire presented in Table 1 to enable a practical application. The criteria are adjusted to be used as questions. Possible answers are provided in column *Dimension*. The table is structured into three categories. *Mandatory* criteria are knock-out questions where a single *No* indicates that FaaS is not an option.

Cost-Efficiency criteria impact the billing and are therefore relevant from a business perspective. Since FaaS is the first *real* pay-as-you-go billing model, evaluating the potential cost is a significant factor.

The impacts of *individual* criteria depend on the specific situation. Regardless of the answers given to these questions, an implementation with FaaS is not immediately prevented. Each developer or organization has to decide individually how important these aspects are. If, for example, vendor independence is important, it generally makes FaaS less suitable. But if the disadvantages can be dealt with, it might nevertheless be possible to implement a functionality using FaaS.

This questionnaire helps assess the potential of implementing functionalities as serverless functions by substantiating the decision process. The criteria are intended to be assessed on the basis of a predefined functionality. However, finding the right scope for a functionality or decide whether several functionalities could be fused into one serverless function is another important topic which has to be addressed in research.

5 Related Work

Criteria to consider in the decision process for or against using FaaS have so far not been discussed in combination but individually as suitable for the respective purpose. To our knowledge, this is the first work collecting these criteria in a catalog. The only comparable approach is a flow chart created by Bolscher as part of his Master's thesis [6], but it considers all criteria to be yes or no criteria.

Research on decision making in the broader topic of cloud computing, however, is already a mature field of research. The main focus is often on multiple-criteria decision making to select a specific cloud service or cloud provider [14]. In comparison to this, our work has a considerably smaller scope and is focused only on a single outcome, namely whether or not to use FaaS. It could however be combined with multiple-criteria decision making to select a suitable deployment model for an application.

6 Conclusion and Outlook

This paper presents a catalog of criteria which can be practically used with the presented questionnaire for the decision of whether to implement a functionality as a serverless function or not. To evaluate their comprehensiveness and usefulness, in future work the criteria should be applied to an actual use case or be empirically validated with the help of practitioners. Furthermore, specific criteria are worth to be considered in more detail such as the computing resource efficiency or the impact of different workload types.

References

1. Aditya, P., Akkus, I.E., Beck, A., Chen, R., Hilt, V., Rimac, I., Satzke, K., Stein, M.: Will Serverless Computing Revolutionize NFV. Proceedings of the IEEE 107(4), 667–678 (2019)
2. Adzic, G., Chatley, R.: Serverless Computing: Economic and Architectural Impact. In: Proceedings of 11th Joint Meeting on Foundations of Software Engineering. ACM, Paderborn, Germany (2017)
3. Albuquerque Jr, L.F., Ferraz, F.S., Oliveira, R.F.A.P., Galdino, S.M.L.: Function-as-a-Service X Platform-as-a-Service: Towards a Comparative Study on FaaS and PaaS. In: The Twelfth International Conference on Software Engineering Advances (ICSEA). p. 217. IARIA (2017)
4. Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., Ishakian, V., Mitchell, N., Muthusamy, V., Rabbah, R., Slominski, A., Suter, P.: Serverless Computing: Current Trends and Open Problems. In: Research Advances in Cloud Computing, pp. 1–20. Springer Singapore (2017)
5. Baldini, I., Cheng, P., Fink, S.J., Mitchell, N., Muthusamy, V., Rabbah, R., Suter, P., Tardieu, O.: The serverless trilemma: function composition for serverless computing. In: Proceedings of the 2017 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software - Onward! 2017. ACM Press (2017)
6. Bolscher, R.: Leveraging Serverless Cloud Architectures. Master's thesis, University of Twente (2019)
7. Eivy, A.: Be Wary of the Economics of "Serverless" Cloud Computing. IEEE Cloud Computing 4(2), 6–12 (2017)
8. Elgamal, T., Sandur, A., Nahrstedt, K., Agha, G.: Costless: Optimizing Cost of Serverless Computing through Function Fusion and Placement. In: 2018 IEEE/ACM Symposium on Edge Computing (SEC). pp. 300–312. IEEE (2018)
9. Eurostat: Cloud computing services used by more than one out of four enterprises in the eu. Online (2018), https://ec.europa.eu/eurostat/documents/2995521/9447642/9-13122018-BP-EN.pdf
10. van Eyk, E., Iosup, A., Seif, S., Thömmes, M.: The SPEC Cloud Group's Research Vision on FaaS and Serverless Architectures. In: Proceedings of the 2nd International Workshop on Serverless Computing. pp. 1–4. WoSC '17, ACM, New York, NY, USA (2017)
11. van Eyk, E., Toader, L., Talluri, S., Versluis, L., Uta, A., Iosup, A.: Serverless is More: From PaaS to Present Cloud Computing. IEEE Internet Computing 22(5), 8–17 (2018)
12. Fehling, C., Leymann, F., Retter, R., Schupeck, W., Arbitter, P.: Cloud Computing Patterns. Springer Vienna (2014)
13. Hellerstein, J.M., Faleiro, J., Gonzalez, J.E., Schleier-Smith, J., Sreekanti, V., Tumanov, A., Wu, C.: Serverless Computing: One Step Forward, Two Steps Back. In: 9th Conference on Innovative Data Systems Research (CIDR) (2019)
14. Lee, S., Seo, K.K.: A hybrid multi-criteria decision-making model for a cloud service selection problem using BSC, fuzzy delphi method and fuzzy AHP. Wireless Personal Communications 86(1), 57–75 (2015)
15. Leitner, P., Wittern, E., Spillner, J., Hummer, W.: A mixed-method empirical study of Function-as-a-Service software development in industrial practice. Journal of Systems and Software 149, 340–359 (2019)

16. Lloyd, W., Ramesh, S., Chinthalapati, S., Ly, L., Pallickara, S.: Serverless Computing: An Investigation of Factors Influencing Microservice Performance. In: 2018 IEEE International Conference on Cloud Engineering (IC2E). pp. 159–169. IEEE (2018)
17. Lynn, T., Rosati, P., Lejeune, A., Emeakaroha, V.: A Preliminary Review of Enterprise Serverless Cloud Computing (Function-as-a-Service) Platforms. In: 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom). pp. 162–169. IEEE (2017)
18. Malawski, M., Figiela, K., Gajek, A., Zima, A.: Benchmarking Heterogeneous Cloud Functions. In: Heras, D.B., Bougé, L. (eds.) Euro-Par 2017: Parallel Processing Workshops. pp. 415–426. Springer International Publishing (2018)
19. Manner, J., Endreß, M., Heckel, T., Wirtz, G.: Cold Start Influencing Factors in Function as a Service. In: 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion, 4th Workshop on Serverless Computing (WoSC). IEEE (2018)
20. McGrath, G., Brenner, P.R.: Serverless Computing: Design, Implementation, and Performance. In: 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW). pp. 405–410. IEEE (2017)
21. McGrath, G., Short, J., Ennis, S., Judson, B., Brenner, P.: Cloud Event Programming Paradigms: Applications and Analysis. In: 2016 IEEE 9th International Conference on Cloud Computing. pp. 400–406. CLOUD, IEEE (2016)
22. Mohanty, S.K., Premsankar, G., di Francesco, M.: An evaluation of open source serverless computing frameworks. In: 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom). pp. 115–120. IEEE (2018)
23. Spillner, J., Al-Ameen, M., Boruta, D.: Serverless literature dataset. Online (2019), https://zenodo.org/record/3517819
24. Yussupov, V., Breitenbücher, U., Leymann, F., Müller, C.: Facing the unplanned migration of serverless applications. In: Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing - UCC'19. pp. 273–283. ACM Press, New York, NY, USA (2019)

All links were last followed on January 10, 2020.

Managing Consent in Workflows under GDPR

Saliha Irem Besik and Johann-Christoph Freytag

Humboldt-Universität zu Berlin, Department of Computer Science,
Unter den Linden 6, 10099 Berlin, Germany
{besiksal,freytag}@informatik.hu-berlin.de

Abstract. The European Union General Data Protection Regulation (GDPR) defines the principles to be met by organizations when processing personal data in order to guarantee data privacy. According to GDPR, consent is required for establishing a legal basis for processing personal data, if there are no other legal grounds for the processing. Besides any identifiable "natural" person, also known as data subject, has the right to withdraw the given consent to process his or her personal data at any time. It is the organization's responsibility to ensure consent and its revocation to demonstrate its compliance with GDPR. With respect to GDPR compliance, organizations can benefit from workflows as they might be used to ensure that consent is obtained before processing personal data. This paper addresses how to enable organizations to manage consent and revocation through their workflows.

Keywords: Data Privacy · General Data Protection Regulation (GDPR) · Consent · BPMN · Business Process Compliance

1 Introduction

The European Union General Data Protection Regulation (GDPR) limits the processing personal data unless it is explicitly allowed by law, or the data subject has consented to the processing [GDPR, Article 6]. In addition to this, the data subject shall have the right to withdraw his or her consent at any time [GDPR, Article 7(3)]. Organizations dealing with personal data of European Union citizens must be able to provide a proof of validity of obtained consent and revocation.

The principle of Privacy by Design (PbD) advocates that privacy should be considered as a first class citizen in the technology design and should be proactively embedded. In order to support PbD, organizations can take advantage of workflows by checking compliance of their workflow models with GDPR during design time. One of the significant benefits of using workflows is that it enables to capture how data is transmitted for what purpose at the conceptual level. We use Business Process Model and Notation (BPMN) to model workflows as it is a de-facto standard for business process modeling.

In this paper, we analyze the consent and its revocation under GDPR. Based on this analysis, we propose design patterns to integrate consent and revocation features in BPMN-based workflows.

J. Manner, S. Haarmann (Eds.): 12^{th} ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

S.I. Besik and J.C. Freytag

The structure of the rest of the paper is as follows: Section 2 briefly presents our understanding of the consent and revocation under GDPR. Section 3 discusses the concept of workflows in our research. Section 4 gives an overview of our proposed approach. Section 5 gives a running example in the clinical domain to show the applicability of our approach. Section 6 reviews related works. Finally, Section 7 concludes this paper and discusses our future work and perspectives.

2 Consent and Revocation under GDPR

Consent can be defined as "any freely given, specific, informed and unambiguous [...] clear affirmative action" by which a data subject agrees to the processing of his or her personal data [GDPR, Article 4]. Some data operations are lawful only if the data subject has given consent to this processing [GDPR, Article 6, §1(a)].

Organizations need to determine whether their data operations require consent to be lawful. We use the term *Consent Policy* to define the statements to declare whether a data operation requires a consent. In our article [1], we give a formal definition for *Consent Policy* in Definition 1.

Definition 1 (Consent Policy) *A Consent Policy CP contains policies which are represented as 2-tuples cp = (purpose, requiresConsent), where:*

- *purpose is the reason for which data is collected, used, or disclosed;*
- *requiresConsent* $\in$ {*true, false*} *specifies whether data processing requires consent or not.*

For instance, newborn hearing screening requires an explicit consent to be legitimate. However, an emergency case does not require consent as it is a subject of "vital interest" which means being necessary to protect someones life. Thus, CP contains policies *(newborn-hearing-screening, true)* and *(vital-interest, false)* accordingly.

For consent to be informed and valid, the data subject must be aware at least of the purposes of the processing and the identity of the data controller who determines the purposes and means of the processing [GDPR, Recital 42]. As an example, assume Hospital X wants to use personal data of its patients for newborn hearing screening. Hospital X can use the following statement to inform its patients to obtain consent *"We, as Hospital X, use your personal data for the purpose of newborn hearing screening."*

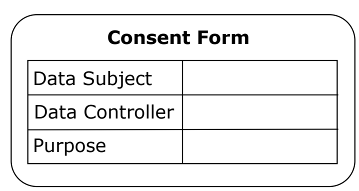


Fig. 1. Consent Form.

We model a *Consent Form* (illustrated in Figure 1) that retains the minimum required information to be valid. *Consent Form* can be elaborated with

Managing Consent in Workflows under GDPR 9

additional information such as the duration of the consent to give data subjects more control.

A data subject has the right to withdraw his or her consent at any time [GDPR, Article 7, §3]. Organizations are obliged to take appropriate actions to handle revocation. They have to stop any ongoing process instances which are affected by revocation. They should also delete the personal data, if the personal data is not used by any other purpose and becomes unnecessary after revocation.

3 Workflows: Platform to support GDPR Compliance

Data privacy, in general, focuses on how personal data should be handled. In order to ensure data privacy in BPMN-based workflows, we work on different ways of data handling supported in BPMN [2]. Data is represented in BPMN via data object, data store, or message elements[1] (shown in the left-hand side of Figure 2). In order to check the privacy compliance of a workflow for each of its data operation, it is essential to know explicitly which personal data is accessed for which purpose. We expect organizations to provide this information via semantic text annotations. Right-hand side of Figure 2 illustrates how we semantically annotate the data operations, where *purpose* refers to the purpose of accessing data and *attribute-name* refers to a set of attribute names of data which are accessed. For example, $< marketing, \{name, age\} >$ annotation means *name* and *age* data attributes are accessed for the purpose of *marketing*.

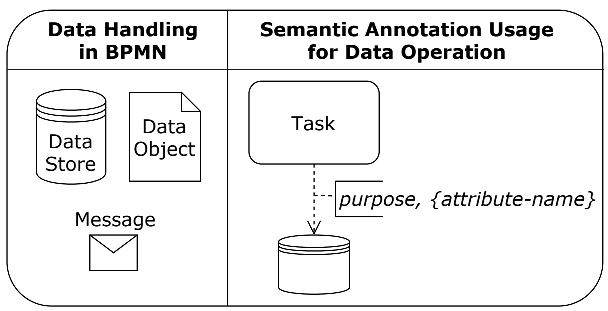


Fig. 2. Data in BPMN.

4 Approach

Our idea how to manage consent and revocation is to propose design patterns that are integrated into BPMN-based workflows. As we explained in Section 2,

[1] https://www.omg.org/spec/BPMN/2.0/

some data operations are legitimate only with the consent of a data subject. As a first step, we determine the data operations in a given workflow. Section 3 lists different means to handle data in BPMN which are via data object, data store, or message elements. We check these BPMN elements in a given workflow. We assume that data operations are all semantically annotated and semantic annotations include the purpose of the data operations.

After determining data operations in a given workflow, we determine whether these data operations require consent according to a given *Consent Policy* (Definition 1). If there is a data operation requiring explicit consent and if there is no consent obtained before that data operation, there is a potential privacy violation. Our idea is adding a checking consent step beforehand to remove this potential violation. For this purpose, we designed a consent pattern which is shown in Figure 3. *Data Controller* asks the *Data Subject* for consent and the *Data Operation* is executed when consent is granted. Otherwise, the process terminates. *Consent Form* is modeled as it is shown in Figure 1, it contains the identity of *Data Controller* and the purpose of the *Data Operation*.

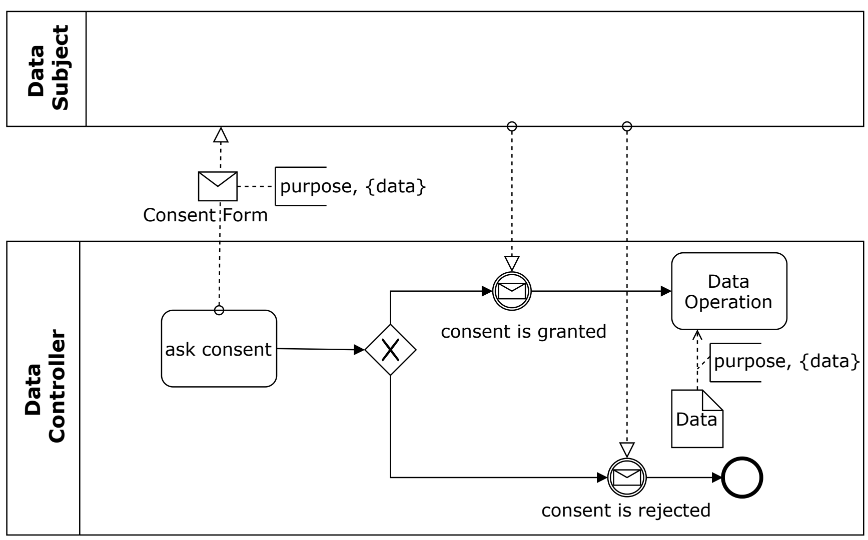


Fig. 3. Consent Pattern.

We address the question "when to obtain consent?". We consider to add consent pattern as late as possible which means to add it just before the concerned data operation. In this way, we intend to prevent the potential situations where consent is obtained yet never used. We argue that consent should be obtained only when it is needed. We raise also the question "what if there are multiple purposes within a given workflow?". When there are multiple purposes,

Managing Consent in Workflows under GDPR 5

consent should be given for all of them [GDPR, Recital 42]. Our strategy to handle multiple purposes is first analyzing whether data operations with different purposes always follow each other. When they follow each other, we create one aggregated *Consent Form* including all their purposes. If data operations do not always occur together, we create separate consent patterns for each of these data operations. The reason behind is again related to prevent the potential situations where consent is obtained yet never used. Creating separate consent patterns might increase the complexity of the workflows in terms of readability. However, a consent pattern can also be designed as a BPMN sub-process which provides a more compact view. Thus, we might increase the readability of the workflow.

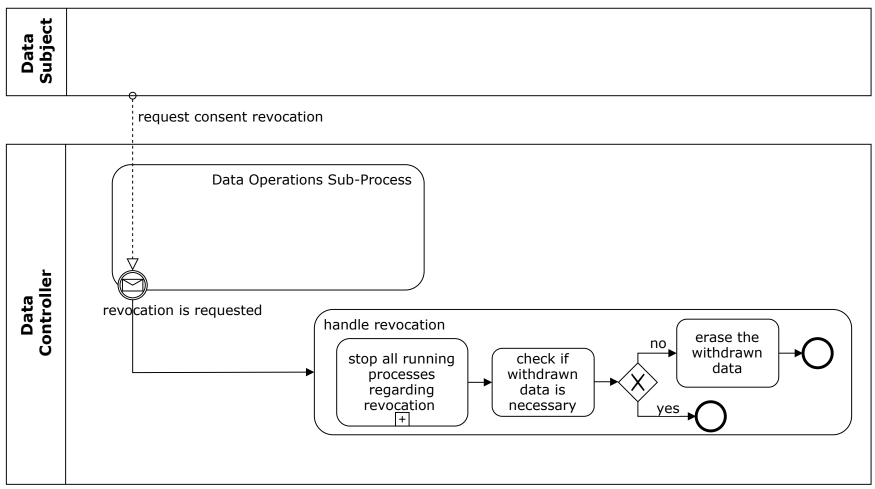


Fig. 4. Revocation Pattern.

Figure 4 shows our design pattern to handle revocation. We created a sub-process which includes all data operations because the consent withdrawal can be related to any of the data operations. We also added a *"handle revocation"* step for the created sub-process. *Handle revocation* task is triggered by withdrawal request by a data subject. Handling revocation implies to stop any ongoing process instances affected by the withdrawal request of the data subject and to erase the personal data if the personal data is not necessary anymore.

We developed both consent and revocation patterns in a way that no additional BPMN symbol is required. In this way, our approach can be easily applied to existing BPMN-based workflows.

5 Running Example

We consider Newborn Hearing Screening (NHS) procedure in Germany as a running scenario in order to illustrate our methodology. It is an optional procedure which requires the explicit consent of at least one of the parents or guardians of the newborn babies. After carrying out NHS, according to the result of the screening pediatrician either applies treatment or conducts further research. Processing personal data for the purpose of research also requires consent to be lawful. Figure 5 shows a BPMN diagram for our scenario. Take note that this BPMN diagram is not GDPR-aware which means there is no consent or revocation control.

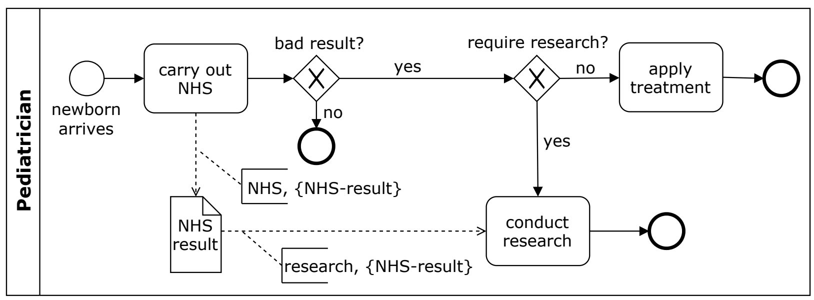


Fig. 5. Newborn Hearing Screening Diagram without Consent and Revocation.

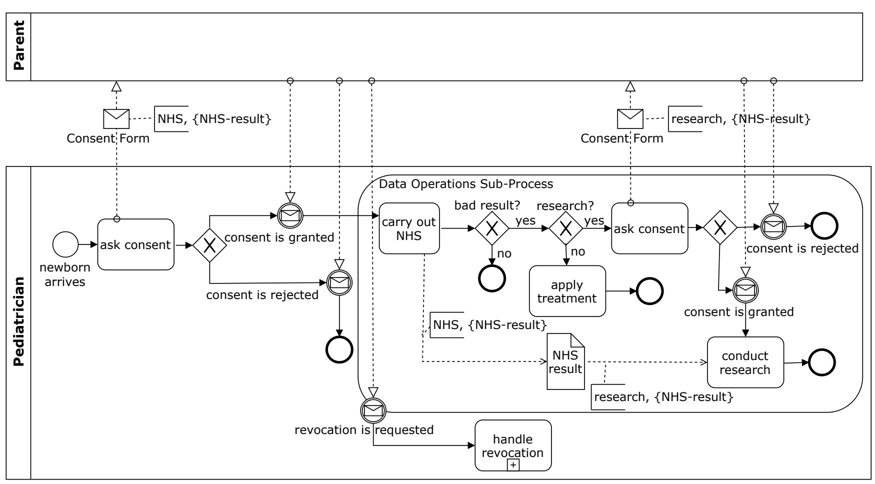


Fig. 6. Newborn Hearing Screening Diagram with Consent and Revocation.

Managing Consent in Workflows under GDPR 57

Figure 6 illustrates the BPMN diagram for newborn hearing screening which includes consent and revocation controls. In our scenario, there are two data operations with two different purposes which are NHS and research. We add separate consent patterns for each of these data operations because when the processing has multiple purposes, consent should be given for all of them. In our scenario, pediatrician do not always conduct research. Therefore, we do not ask consent for research and NHS at the same time. In order to manage revocation, we create the *Data Operations Sub-Process* including both data operations and we add handle revocation pattern for this sub-process. *Handle revocation* task is triggered by a revocation request by a parent.

6 Related Work

Granting and revoking consent effectively has been the focus of several research efforts over the last years. In the pre-GDPR era, one of the pioneers in this field was Ensuring Consent and Revocation (EnCoRe) research project. Within this project, one of the goals was to provide dynamic and granular options for consent and revocation in system design [3][4]. They also provide conceptual modeling for privacy policies with consent and revocation requirements [5]. Their understanding of consent differs from ours as they do not reflect on the requirements and obligations based on GDPR. Also, they have no work regarding the use of workflows to handle consent and revocation. [6] proposes the idea of the alignment of workflows with consent management. Their aim, however generating the letter of consent documents based on existing workflows. In the literature, there are also studies where BPMN is extended towards security and privacy aspects [7][8]. However, these works do cover neither consent nor revocation. [9] presents a set of design patterns as business process models which enables organizations to tackle GDPR constraints. Their work can be considered as a guide to achieve GDPR compliance for an organization. Our work considerably differs from their work because we aim to transform the existing non-privacy-aware business process models into privacy-aware ones.

7 Conclusion and Outlook

We are convinced that it is fundamental to incorporate consent and revocation controls within the workflows of organizations that handle personal data to ensure their compliance with GDPR during the design time. In this paper, we have presented our approach to adapt the BPMN-based workflows with the consent and revocation concepts under GDPR. As future work, we would like to work on how to automatically generate the consent forms by using the existing workflows and automatically transform the existing workflows into the ones which handle the consent and revocation efficiently. We are also interested in analyzing the optimality of our approach.

References

1. Saliha Irem Besik and Johann-Christoph Freytag. A formal approach to build privacy-awareness into clinical workflows. *SICS Software-Intensive Cyber-Physical Systems*, pages 1–12, 2019.
2. Saliha Irem Besik and Johann-Christoph Freytag. Ontology-based privacy compliance checking for clinical workflows. In *Proceedings of the Conference on "Lernen, Wissen, Daten, Analysen", Berlin, Germany, September 30 - October 2, 2019*, pages 218–229, 2019.
3. Marco Casassa Mont, Siani Pearson, Gina Kounga, Yun Shen, and Pete Bramhall. On the management of consent and revocation in enterprises: setting the context. *HP Laboratories, Technical Report HPL-2009-49*, 11, 2009.
4. Ioannis Agrafiotis, Sadie Creese, Michael Goldsmith, and Nick Papanikolaou. Reaching for informed revocation: Shutting off the tap on personal data. In *IFIP PrimeLife International Summer School on Privacy and Identity Management for Life*, pages 246–258. Springer, 2009.
5. Marco Casassa Mont, Siani Pearson, Sadie Creese, Michael Goldsmith, and Nick Papanikolaou. A conceptual model for privacy policies with consent and revocation requirements. In *IFIP PrimeLife International Summer School on Privacy and Identity Management for Life*, pages 258–270. Springer, 2010.
6. Nils Gruschka and Meiko Jensen. Aligning user consent management and service process modeling. In *GI-Jahrestagung*, pages 527–538, 2014.
7. Wadha Labda, Nikolay Mehandjiev, and Pedro Sampaio. Modeling of privacy-aware business processes in BPMN to protect personal data. In *Proc. of the 29th Annual ACM Symposium on Applied Computing*, pages 1399–1405. ACM, 2014.
8. Alfonso Rodríguez, Eduardo Fernández-Medina, and Mario Piattini. A BPMN extension for the modeling of security requirements in business processes. *IEICE transactions on information and systems*, 90(4):745–752, 2007.
9. Simone Agostinelli, Fabrizio Maria Maggi, Andrea Marrella, and Francesco Sapio. Achieving gdpr compliance of bpmn process models. In *International Conference on Advanced Information Systems Engineering*, pages 10–22. Springer, 2019.

The Ultimate Comparison Framework

Oliver Kopp

IPVS, University of Stuttgart, Germany
kopp@informatik.uni-stuttgart.de

Abstract. When deciding for a tool, there are differnt tools to choose from available. Researches come up with scientific criteria to compare different tools and spend huge effort to run the comparison. The presentation of the comparison results is an open issue. The "Ultimate Comparison Framework" is one solution enabling a) collection of evaluation data using markdown and b) presentation of the data set in a web-app.

1 Introduction

The accumulation of knowledge is an essential condition for a field to be scientific and to develop [3]. In the field of information systems research in general and business process management in particular, literature reviews are conducted to harvest the body of knowledge [6, 7]). There is still lack of sharing analysis results, especially in the field of business process management (cf. Recker and Mendling [7]). The Ultimate Comparison Framework is an approach to fill this gap: It supports publishing reviewing results in a open way. It further supports updating the research results by established collaborative software engineering techniques such as GitHub pull requests [2]. The current focus is on comparison of tools and technologies, but not limited to it.

The development of the Ultimate Comparison Framework was driven by a) sustaining search results in the context of finding the most fitting tool for a task and b) offering a framework for comparitive studies crafted by students in the context of their software engineering trainings (cf. "Fachstudie" [5]).

The framework is called "Ultimate Comparision Framework", because it offers creation of "Ultimate Comparisions" and is not a single comparison of different tools. The name "Ultimate Comparision" stems from the claim that the framework is easy to use (covering creation and maintenance of data as well as reading data) and will be used by many research to present their research results. This paper is a first step into this direction by making it known to a broader community.

Users of the framework are a) researchers wanting to sustainn their survey results, b) researches investigating other surveys, and c) industry users interested in introducing a new tool or framework and aiming for a scientifically-grounded comparison of existing work.

Already published "ultimate comparisions" include: Comparison of Cloud Deployment and Management Tools[1], Comparision of IoT platforms[2], Comparison of time-

[1] https://ultimate-comparisons.github.io/ultimate-deployment-tool-comparison/
[2] https://ultimate-comparisons.github.io/ultimate-IoT-platform-comparison/

J. Manner, S. Haarmann (Eds.): 12^{th} ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-XXXX

series databases[3], Comparison of message brokers[4], Comparison of graph libraries for JavaScript[5], Comparison of web-based IDEs[6], Comparision of literature management software[7], and Comparison of LaTeX building helpers[8]. The list of available comparisions is constantly updated at `https://ultimate-comparisons.github.io/`.

The idea stems from the PaaSfinder Web Application [4] (available at `https://paasfinder.org/`). In contrast to PaaSfinder, the Ultimate Comparison Framework is hosted as a Web site and stores its data in Markdown. It is the first tool basing on plain-text (Markdown) for data storage and on GitHub for data collection and data presentation. Its source is available at `https://github.com/ultimate-comparisons/ultimate-comparison-BASE`.

This paper demos the framework by showing the Ultimate Comparison of Open Source Time-Series-Databases [1]. We explain the reader-facing interface (Sect. 2) and the contributor-facing interface (Sect. 3). After sketching the implementation (Sect. 4), we provide a short summary and outlook (Sect. 5).

2 Reader-facing Interface

A reader sees a criteria overview (Fig. 1) and a table listing all tools (Fig. 2). The criteria overview is automatically generated from the criteria configuration (Sect. 3). Besides using common filtering and sorting techniques, a user can also configure visibility as well as export to LaTeX. The export to LaTeX is important to embed the results in a scientific technical report such as publications or a student thesis. The examples are based on the Ultimate Comparision of time-series databases.

3 Contributor-facing Interface

In the current deployment, the Ultimate Comparison uses GitHub as hosting platform. The typical GitHub flow[9] is in place. Each dataset of a tool is stored in a separate Markdown file. Editing of one such a file is shown in Fig. 3. The criteria as well as the possibles values are configured an ultimate comparison (using a YAML configuration file). Thus, the framework allows for defining arbitrary criteria names. Possible data types are text, enums, and numbers. For instance, the code license can be enumerated and results of performance meassurements can be provided. Each criteria is input as Markdown heading. In case of text, the text comes below a the heading (e.g., "`- Apache-2.0`"

[3] `https://tsdbbench.github.io/Ultimate-TSDB-Comparison/`
[4] `https://ultimate-comparisons.github.io/ultimate-message-broker-comparison/`
[5] `https://ultimate-comparisons.github.io/ultimate-graphframework-comparison/`
[6] `https://ultimate-comparisons.github.io/ultimate-webIDE-comparison/`
[7] `https://ultimate-comparisons.github.io/ultimate-reference-management-software-comparison/`
[8] `https://ultimate-comparisons.github.io/ultimate-latex-makers-comparison/`
[9] `https://guides.github.com/introduction/flow/`

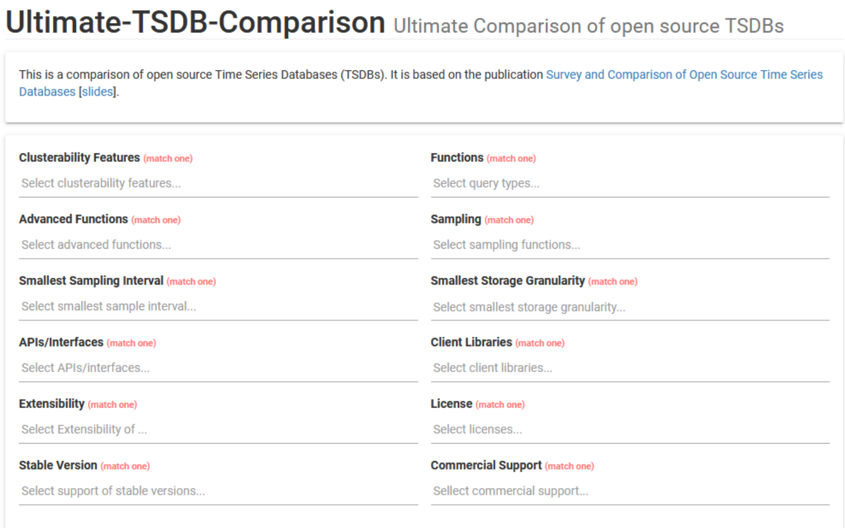


Fig. 1. Criteria overview

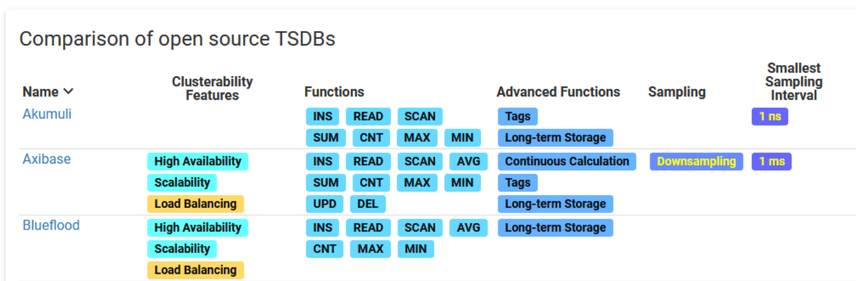


Fig. 2. Table of tools (bottom omitted)

below "`## License`"). In case of an enum, all matching enums are listed as markdown list. In case of numbers, the number is given below the heading.

To setup an ultimate comparision, basic knowledge of GitHub pages and the usage of CI/CD tools is necessary. Furthermore, writing YAML is an essential software engineering skill. To input data into an ultimate comparision, the knowledge of using GitHub as well as entering Markdown are necessary.

4 Implementation

The Web interface is implemented using Angular. The data is converted from Markdown to JSON using Java on a CI server (currently TravisCI). The JSON is read by the Web

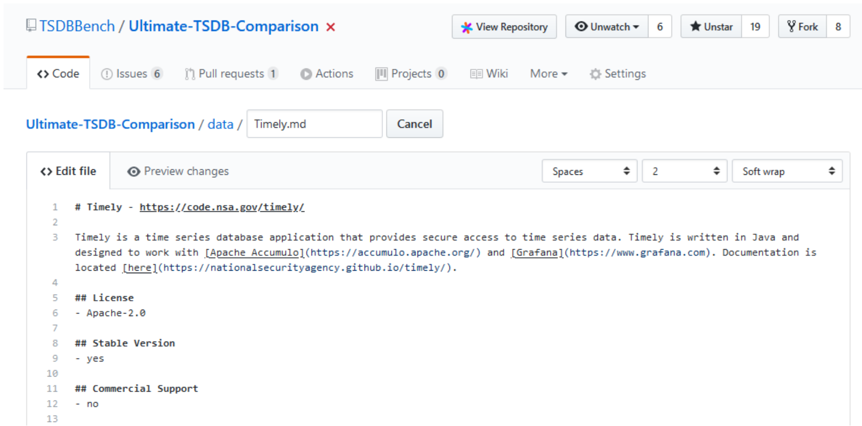


Fig. 3. Editing data using GitHub

application running on the client side in the browser. The Web interface is hosted in the branch “gh-pages” of the respective comparison and offered to users by the GitHub Pages offering[10].

5 Conclusion and Outlook

This paper presented the Ultimate Comparison Framework offering a collaborative comparison framework. It stores the data in markdown files enabling the usage of arbitrary text editors to add or modify data. The the data is rendered as static HTML page showing all criteria, the evaluated tools, and the fulfillment of each criterion in a table.

The next development ideas are a) offering a browser-based user interface to input data (instead of relying on GitHub) and b) using Wikipedia tables as data input, either as data source for displaying or as data source for synchronizing the local Markdown files. Future work includes measurement of the time required to publish and maintain results in comparison to other approaches such as scentific publications or tables in WikiPedia articles.

Acknowledgments We want to thank Stefan Kolb for the idea and inspiration of this tool Further, we want to thank Andreas Bader, Armin Hüneburg, and Christoph Kleine for discussions on the UI and driving the implementation. This work is partially funded by the Federal Ministry for Economic Affairs and Energy (BMWi) projects *Industrial Communication for Factories* (01MA17008G), *NEMAR* (03ET4018), and *SmartOrchestra* (01MD16001F).

[10] `https://pages.github.com/`

References

1. Bader, A., et al.: Survey and comparison of open source time series databases. In: BTW2017 Workshops. GI e.V. (2017)
2. Gousios, G., Pinzger, M., van Deursen, A.: An exploratory study of the pull-based software development model. In: Proceedings of the 36th International Conference on Software Engineering - ICSE 2014. ACM Press (2014)
3. Hunter, J.E., Schmidt, F.L., Jackson, G.B.: Meta-analysis: Cumulating research findings across studies. Educational Researcher 15(8), 20 (oct 1986)
4. Kolb, S.: On the Portability of Applications in Platform as a Service. Ph.D. thesis, University of Bamberg, Germany (2019)
5. Ludewig, J.: Erfahrungen bei der lehre des software engineering. In: Software Engineering im Unterricht der Hochschulen. dpunkt.verlag (2009)
6. Paré, G., Trudel, M.C., Jaana, M., Kitsiou, S.: Synthesizing information systems knowledge: A typology of literature reviews. Information & Management 52(2), 183–199 (mar 2015)
7. Recker, J., Mendling, J.: The state of the art of business process management research as published in the BPM conference. Business & Information Systems Engineering 58(1), 55–72 (nov 2015)

All links were last followed on January 17, 2020.

Index