Event-driven Monitoring of Business Processes in Microservice Architecture

Niklas Sprenger[1], Bastian Schierbaum[1], and Jonas Sorgalla[2]

[1] Vanderlande Industries GmbH, Dortmund, Germany
{niklas.sprenger | bastian.schierbaum}@vanderlande.com
[2] University of Applied Sciences and Arts Dortmund, Dortmund, Germany
jonas.sorgalla@fh-dortmund.de

Abstract

Increasingly complex software systems and the business processes behind them can quickly become intransparent and difficult for users to understand. Especially when processes fail or are delayed, it is important to provide users with a simple and fast way of monitoring so that they can react accordingly. In our talk, we use the example of the logistics company Vanderlande to show how successful process monitoring can be realized by combining BPMN and an event-based Microservice Architecture.

1 Introduction

Vanderlande is a global logistics company that develops and distributes automated customer solutions in the areas of warehouse, parcel & postal, and airport. The developed hardware and software is used to automate various logistics processes and their sub-parts for customers. A typical example of a logistic process would be the execution of an outbound order. Although automated logistics processes mean an enormous gain in efficiency, errors such as stuck conveyor belts or an object falling off the belt occur during process execution and can not totally be prevented due to the mechanical nature of the system. These errors need to be fixed manually and error tracing often results in tedious and very time-consuming working through a large amount of text-based log files. Furthermore, the person charged with error handling needs a high level of system and process knowledge in order to quickly identify and scan the right log files. To address this challenge we present a visual and non-invasive monitoring solution based on the Business Process Modeling Notation (BPMN) [5] utilizing the event-driven nature of the Microservice Architecture (MSA) [4] implemented at Vanderlande.

2 Monitoring Solution

2.1 Microservice Architecture at Vanderlande

The MSA at Vanderlande is tailored based on the different sub-tasks that have to be handled within a logistics process. Since we use our system to control time-sensitive mechatronic systems in logistics processes, we rely on a scalable and asynchronous event-based communication using Apache Kafka[1] which is especially suited for such time-critical scenarios [3]. Following the event-driven architecture pattern [2], each of the microservices sends information about state changes, e.g., finished or failed tasks, as an event using Kafka. Other services subscribed to the event can react accordingly and, thus, push-based service communication chains are established. We utilize this event-based characteristic of our microservice architecture in the following for the graphical monitoring of our business processes.

[1] https://kafka.apache.org/

2.2 Process Monitoring

At Vanderlande each logistic process is modeled using the BPMN for documentation purposes. Our monitoring approach enhances the value of these already existing static models by combining them with the occurring communication events fired in the microservice architecture. We realized our approach in a prototype that generates BPMN diagrams in which activity frames are colored based on the occurrence of predefined events as exemplified in Figure 1. For the prototype, we focused on simple visual feedback of completed activities (green frame) and faulty ones (red frame). For example, in the shown diagram the activities *Create Consignment* and *Create Package* were completed for an order, but the *Allocate Packages* activity encountered a problem.

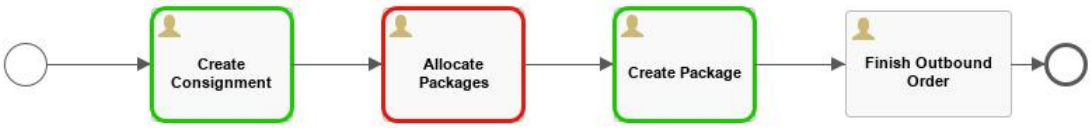

Figure 1: Example process state visualization generated the prototype.

The prototype comprises a backend based on the flowable[2] BPMN engine and an angular frontend. It provides means to store static BPMN models, define the events, and the corresponding messaging topics which the application should react to. Furthermore, the user can specify the dynamic behavior for the BPMN monitoring by configuring the mapping between activities and events, i.e., storing which events from the architecture indicate successful or faulty processing of a certain BPMN activity. Because of the different granularity, i.e., not one but most of the time multiple microservices are involved in the processing of an activity, our prototype allows to define sets of events as success indicators. During execution, the application listens to the specified message topic and maps incoming events during runtime. By doing so, the application shows all running processes and visualizes their current state dynamically.

2.3 Architecture Integration

Figure 2.3 exemplifies the non-invasive characteristic of our prototype in a simplified microservice system in the style of Vanderlande's MSA. Hereby, an external system, e.g., a conveyor belt, pushes information into the system in form of JSON encoded events. In addition, several services also use RESTful HTTP [1] as synchronous communication means for interactions originating from the web-based graphical user interface.

As the examples indicates, our prototype only needs to pull data from the broker. Therefore, it can be easily integrated in new systems, i.e., only changes to the prototype need to be made, but not to any microservices or to the broker.

[2] https://flowable.com/

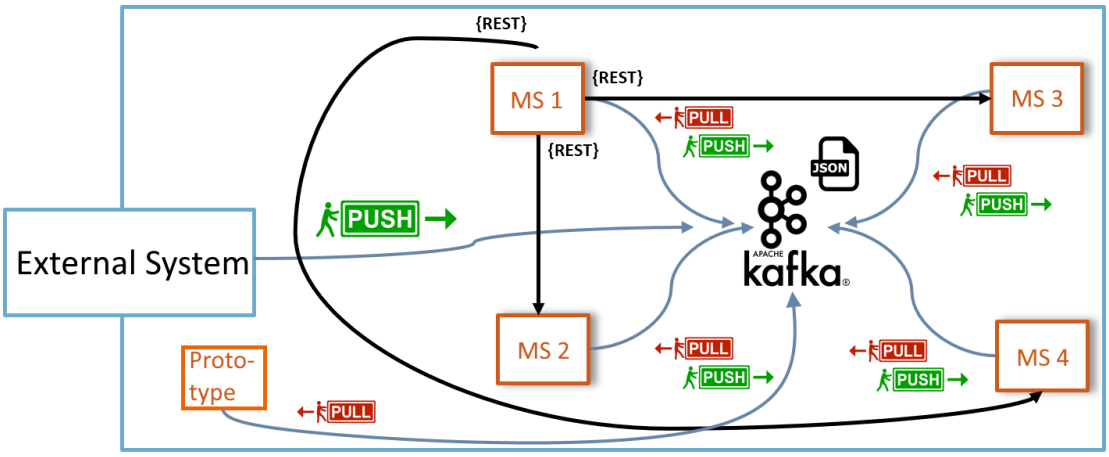

Figure 2: Example of an event-based architecture comprising the prototype application.

3 Conclusion and Future Work

As shown by the prototype, our approach of combining BPMN models and event-based microservice architecture is a feasible and non-invasive approach to monitor the progress of operations based on our defined processes and thus minimize the effort behind the manual error tracing. However, it depends on an event-based communication in the microservice architecture and does not completely replace manual searching, but it does facilitate it.

Based on the prototype, we plan to implement the approach in our production environment. As a positive side effect we will be able to increase the value and, thus, the percieved usefulness of the BPMN models inside Vanderlande. This could also have a positive impact on efforts to keep the models up to date.

References

[1] Roy Fielding. Representational state transfer. *Architectural Styles and the Design of Network-based Software Architecture*, pages 76–85, 2000.

[2] David C. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley Longman Publishing Co., Inc., USA, 2001.

[3] Neha Narkhede, Gwen Shapira, and Todd Palino. *Kafka: the definitive guide: real-time data and stream processing at scale*. " O'Reilly Media, Inc.", 2017.

[4] Sam Newman. *Building Microservices*. O'Reilly Media, 2015.

[5] Object Management Group. *Business Process Model and Notation (BPMN)*. Object Management Group, 2014, Version 2.0.2.