YRKESHÖGSKOLAN
NOVIA

Determination and Implementation of Suitable BPMS Solutions

A journey into a strive for a data driven enterprise

Karl Henrik Bäckström

Degree Thesis for Bachelor of Engineering

Degree Programme in Electrical Engineering and Automation

Vasa 2020

BACHELOR'S THESIS

Author: Henrik Bäckström
Degree Programme: Electrical Engineering and Automation
Specialization: Automation
Supervisors: Erik Englund, Nishant Redekar

Title: Determination and Implementation of Suitable BPMS Solutions

---

Date: May 7,2020 Number of pages: 38

Abstract

This Bachelor's thesis was made on behalf of Wärtsilä in order to investigate and determine suitable ways of implementing solutions of business process management systems (BPMS). This is a theoretical foundation for further development of an overlying business automation platform.

This could establish an audit traceable workflow-platform, interconnecting existing enterprise resource planning (ERP) tools with human processes. Through the use of business process modelling (BPM), graphical modelling tools for defining rules and tasks will be included in order to bridge the gap of the business specialist being the process owner and the technical developer being responsible for the underlying service infrastructure. The approach of implementing such systems is done through an introduction to a set of standards.

During the assessment the available solutions being considered were limited to two. For a proof of concept, a demonstrative frontend was implemented in order to provide a demarcated end-user experience for initiating and interacting with adapted processes. The thesis resulted in an approval for a practical pilot project.

Language: English Key words: BPMS, BRMS, BPA, BPMN, DMN, CMMN, ERP

EXAMENSARBETE

Författare: Henrik Bäckström

Utbildning och ort: El- och Automationsteknik, Vasa

Inriktningsalternativ: Automation

Handledare: Erik Englund, Nishant Redekar

Titel: Determination och implementering av tillämpliga BPMS-lösningar

---

Datum: 7. maj 2020 Sidantal: 38

Abstrakt

Detta examensarbete utfördes för Wärtsilä för att undersöka och bestämma lämpliga lösningar för att implementera affärsprocesshanteringssystem (BPMS). Detta arbete skall bilda en teoretisk grund för framtida utveckling av en överliggande affärsautomations plattform.

Genom detta skall en auditerbar arbetsflödesplattform kunna införas som kopplar samman befintliga verktyg för företagsresursplanering (ERP-system) med mänskliga processer. Med introduktionen av affärsprocessmodellering (BPM) tillkommer grafiska modelleringsverktyg för att definiera regler och uppgifter för att överbrygga klyftan mellan affärsspecialisten som är processägaren och den tekniska utvecklaren som ansvarar för tjänst-infrastrukturen. Detta görs genom en uppsättning av standardiserade medel och metoder.

Under utvärderingen begränsades de beaktade lösningarna till två stycken. För att demonstrera en tillämpning av systemet blev ett webbanvändargränssnitt utvecklat. Dess syfte var att ge en avgränsad användarupplevelse för att starta och behandla de adapterade processerna från de underliggande systemen. Demonstrationen resulterade i ett beviljande av ett praktiskt pilotprojekt.

Språk: engelska Nyckelord: BPMS, BRMS, BPA, BPMN, DMN, CMMN, ERP

OPINNÄYTETYÖ

Tekijä: Henrik Bäckström
Koulutus ja paikkakunta: Sähkö- ja automaatiotekniikka, Vaasa
Suuntautumisvaihtoehto: Automaatiotekniikka
Ohjaajat: Erik Englund, Nishant Redekar

Nimike: Sopivien BPMS-ratkaisujen määrittäminen ja toteutus

Päivämäärä: 7.5.2020 Sivumäärä: 38

Tiivistelmä

Tämä tutkielma on tehty Wärtsilälle. Opinnäytetyön tavoitteena oli tutkia ja määrittää sopivia tapoja toteuttaa liiketoimintaprosessien hallintajärjestelmä (BPMS). Tämä on teoreettinen perusta päällekkäisen liiketoiminta-automaatioalustan edelleen kehittämiselle.

Tavoitteena oli auditoitavan työnkulkualustan käyttöönoton tutkinta, joka yhdistäisi olemassa olevat toiminnanohjausjärjestelmät (ERP) ihmiskeskeisiin prosesseihin. Liiketoimintaprosessien mallinnuksen (BPM) käyttöönoton myötä esitetään graafiset mallinnustyökalut sääntöjen ja tehtävien määrittelemiseksi, jotta voidaan kaventaa kuilua prosessinomistajana toimivan liiketoiminta-asiantuntijan ja palveluinfrastruktuurista vastaavan teknisen kehittäjän välillä. Tämä tapahtuu käyttämällä aihealueen standardeja.

Arvioinnin aikana käytettävissä olevat ratkaisut rajattiin kahteen vaihtoehtoon. Konseptin todistamiseksi loppukäyttäjälle toteutettiin demonstratiivinen käyttöliittymä. Tämä mahdollisti valittujen toiminnallisuuksien toteuttamisen liittyen prosessien aloittamiseen ja vuorovaikutukseen niiden kanssa. Tulos johti käytännön pilottiprojektin hyväksymiseen.

Kieli: englanti Avainsanat: BPMS, BRMS, BPA, BPMN, DMN, CMMN, ERP

Table of contents

1 Introduction. 1
1.1 The commissioner 1
1.2 The thesis topic 1
1.3 Means of solution 1
2 The concept behind business process management systems 2
3 Theory 4
3.1 Standards 4
3.1.1 Business Process Modelling and Notation (BPMN) 4
3.1.2 Decision Model and Notation (DMN) 6
3.1.3 Case Management Model and Notation (CMMN) 8
3.1.4 Predictive Model Markup Language (PMML) 9
3.2 Software architectural concepts 10
3.2.1 Distributed microservices 10
3.2.2 REST 10
3.3 Development and deployment tools 11
3.3.1 Git 11
3.3.2 Apache Maven 11
3.4 Runtime stack 11
3.4.1 Java runtime 12
3.4.2 Java EE 12
3.5 Deployment stack 13
3.5.1 Docker 13
3.5.2 Kubernetes 14
4 Solutions considered during assessment 16
4.1 Red Hat Process Automation Manager 16
4.1.1 KIE-Server 17
4.1.2 Business Central 17
4.1.3 Red Hat Enterprise Application Platform (EAP) 17
4.1.4 OpenShift 18
4.2 Camunda 18
5 Evaluation 20
5.1 Red Hat Process Automation Manager 20
5.2 Camunda 24
5.3 Conclusion 27
6 Implementation 29

6.1 Proof of Concept 29
6.2 Use case 29
6.3 Implementation 30
6.4 Outcome 31
7 Conclusion 32
8 References 35

1 Introduction.

1.1 The commissioner

This thesis work was carried out for the Information Management department at Wärtsilä. Wärtsilä was first founded in 1834 in eastern Finland when a sawmill was established. During the following years, its premises was to be expanded in order to include iron works facilities and then later on renamed to what we know it of as of today. Today, the company has evolved into a global actor in the energy and marine market with an aim to enable the industry transformation into being of 100% renewable energy. In the ongoing strive for change, companies are trying to adapt to data driven approaches, which in turn is where this thesis becomes relevant.

1.2 The thesis topic

The task of this thesis is to research suitable solutions and ways of implementing tools for Business Process Modelling (BPM) or Business Process Management Software (BPMS) for a foundation to build robust process automation. Ultimately, this would allow internal departments to define or describe their needs for process flows in BPMN, a notation standard which would help to create or tie in automation which would remove human error, with the added benefit of proper audit trails for each process taken into account. Part of the challenge is to integrate such systems to make it interconnected in order for automation to become viable in the first place.

1.3 Means of solution

The ideal solution would allow business experts to model and define rules for decisions and then reuse those in definable business processes. The tools for creation of these models should be extendable in such a manner that it is easy to provide a solid framework to cover all bases.

In addition, the system itself should be flexible enough. It must not be too troublesome to incorporate integrations with already existing services. At the same time the defined rules can be reused in other applications by means of communication such as REST-APIs. This would bring additional value even outside the solution in the future.

2 The concept behind business process management systems

Today each and every company has to have a strategy for keeping track of bookkeeping and managing corporate trading agreements and relationships with customers, suppliers and other vendors. Hence, there is a big market for software-based solutions as it comes to aiding companies in different aspects to the core business. This becomes of big importance when a company is growing or operating on a bigger scale across the globe.

There is already a heap of domain specific solutions which provide so called enterprise resource planning (ERP), customer relationship management (CRM) and supply chain management (SCM) solutions. You might have heard of big vendor names or actors in this branch such as SAP, Oracle, Salesforce, Microsoft and so on. (Harmon, 2014)

Nevertheless, this helps, but there are still underlying processes where these solutions play their respective important parts. These processes can usually be characterised by a chain of events along with different mediums and interactions between humans and machines. In this space there lies untapped potential for introducing process automation that could bring a standardised way of interconnecting humans and machines. This also opens possibilities to provide a system overreaching process insight which you would not have had earlier. (Harmon, 2014)

In order to achieve this level of automation, there needs to be a common determinator between the technical system implementation and the actual process owner. In other words, there is a gap to bridge between the actual business process specialists and the developers responsible for delivering such IT-infrastructure. In order to avoid inflexible workflow process implementations and growing technical debt, there is a need to include the process owners in a more flexible way. This is where a set of three standards introduce ways of defining an intermediate design format between system implementation and business specialist. These things will be handled more in detail in the following chapters. (Harmon, 2014)

By relying on these standardised description formats, the models could double as a knowledge store. Big and overreaching processes could be split into smaller ones with accompanying adequate documentation. This could help to gather a definitive process documentation which would otherwise be spread out in different departments or business

units among different employees. Then if the BPMS solution is successfully implemented as the go-to-operation platform, you could gain complete audit traceable logging and reporting capabilities on top of company-wide internal processes. (Harmon, 2014)

From the perspective of the end-users, the solution would come into play in different ways. Generally, there should be a user interface to use for any employee to initiate a process, whether or not it has anything to do with an approval-process. This user interface should also be the go-to tool for handling request reviews alike. When it comes to the creation and management of adapted processes, there should be a separate platform or set of tools which are not visible for the general employee. Therefore, there should be a clear distinction between the actual automation platform development and the end-user experience for utilising the processes themselves.

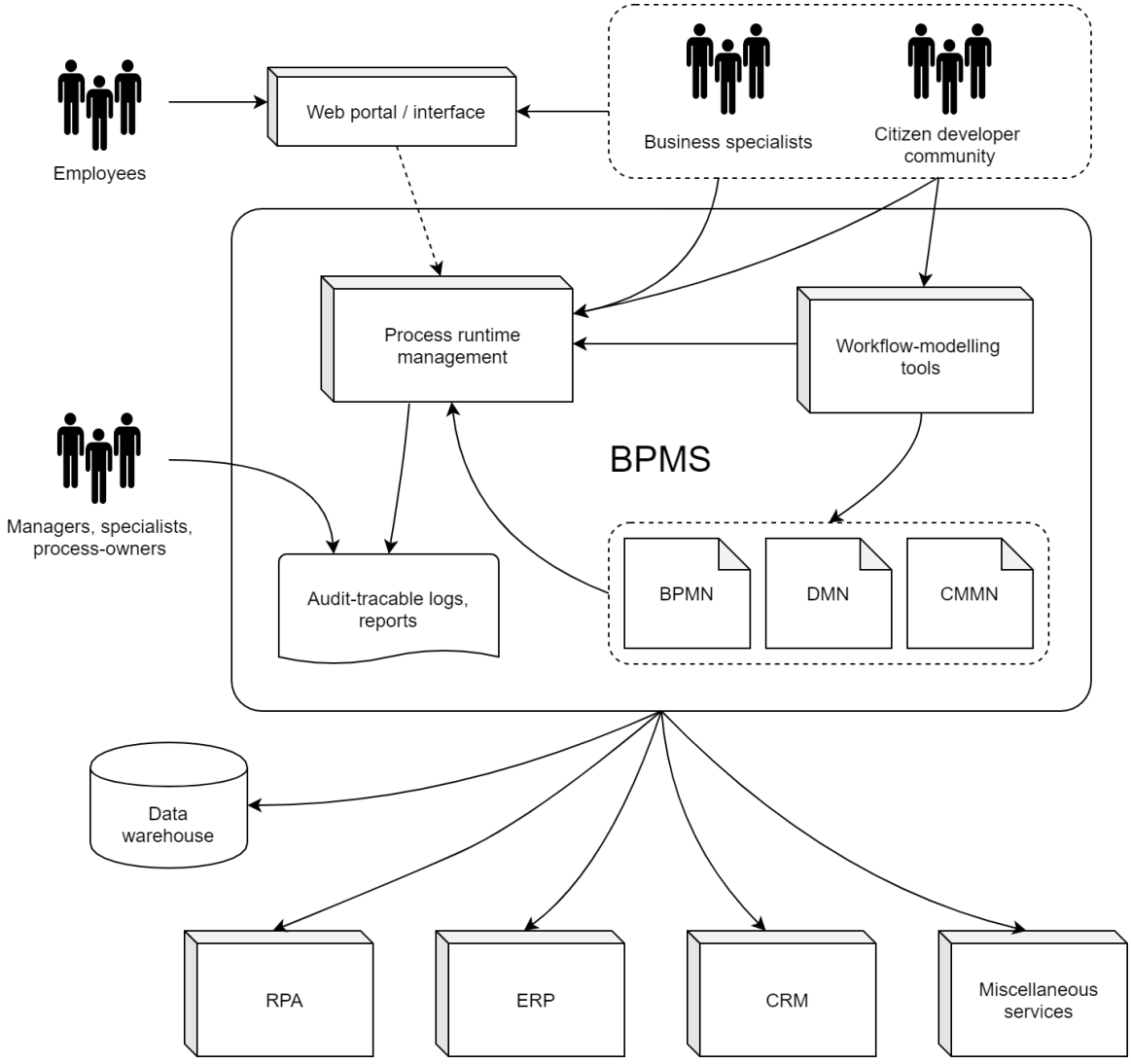

Figure 1: Map of the correlation of concept services and standards with existing services and their respective end-users. Note that the technical developers are involved in all stages in order to provide the infrastructure.

The business process specialists will end up on both sides of the spectrum. The process development flow (creation of low-code model descriptions) and its tools will be aimed for a citizen developer community approach where departments can slowly adapt their processes themselves to suit their respective needs without being limited by a fixed development routine concerning a new feature for an existing platform. Consequently, that would give the flexibility and control of the process where the actual knowledge is at the same time as the problem of indirect ownership of process implementations in code is lifted from the technical developer.

3 Theory

3.1 Standards

There are several standards which will become relevant through this work. Those standards are described in the following sub-sections. One common attribute of all these standards is that they are usually being serialized into Extensible Markup Language (XML), which can be interpreted by other application implementations for interchangeability.

3.1.1 Business Process Modelling and Notation (BPMN)

The idea behind the BPMN standard is to create a notation that is easy enough to understand both from the business specialist's standpoint as well as from the technical developer's standpoint who is to implement that process. In other words, a common way of presenting process descriptions between designer and implementor. Due to the fact that there are a handful of other more or less vaguely related notations which have attempted to partially solve this problem, this is a notation that would try to tie their ideas together into one remaining standard. One of the later versions of BPMN (namely 2.0.1) is also known as ISO/IEC 19510:2013, whereas the latest version is one revision ahead at the time of writing.

From the designer viewpoint this standard provides a graphical flowchart like format which follows an imperative design paradigm. This means that each path from start to end in the described process should be known to a full extent. To attain a format which combines understandability with complex process workflows, BPMN introduces a set of schematics. (Object Management Group, Inc., 2013)

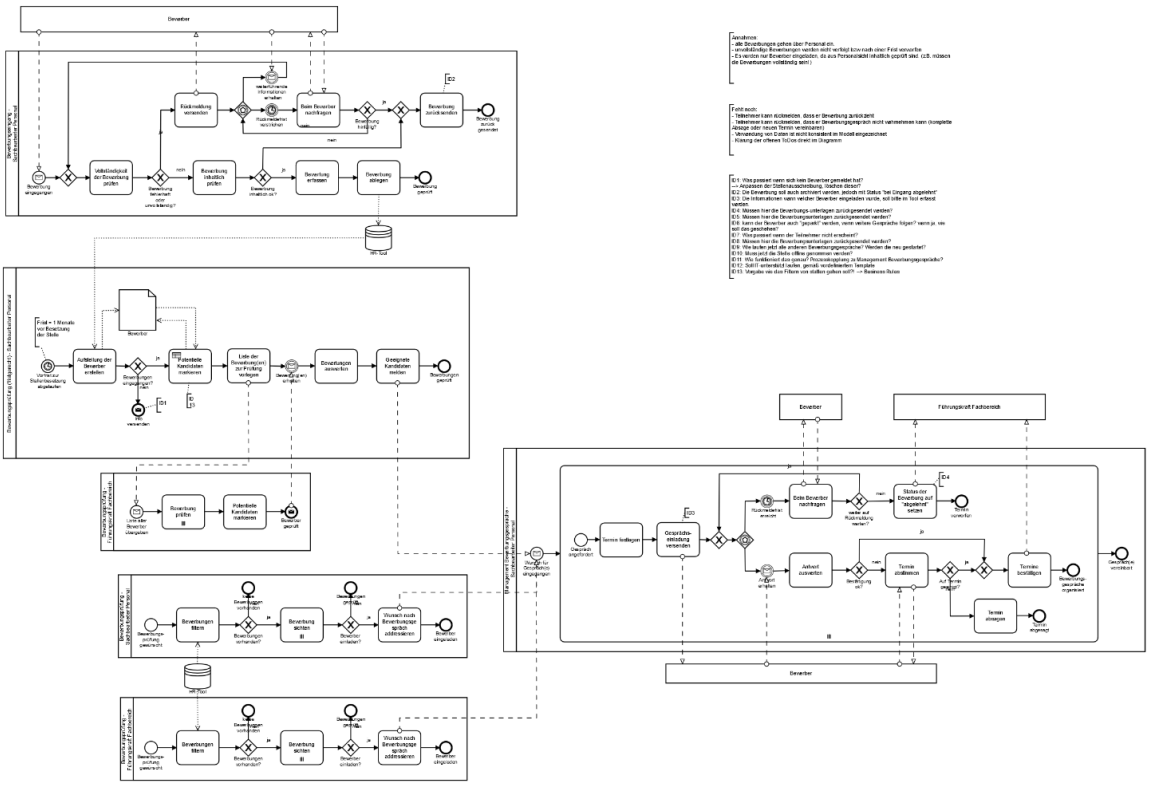

Figure 2: A complex BPMN diagram taken from the sample Visual Studio Code extension project from BPMN.io. (Kiefer, 2019)

The schematics are grouped into the following:

- Flow objects
- Connecting objects
- Swim lanes
- Artifacts

Among the flow objects there are events, activities and gateways. While events are self-explanatory, activities allow you to define a certain task or sub-process which needs to take place in the sequence. These tasks are otherwise used to trigger services, scripts, user tasks, or alternatively, decisions or cases based on the DMN and CMMN standards. Gateways are for routing the workflow sequence. This can be done in several different configurations, for instance, exclusive, inclusive, parallel branching or in alternative event-based gateways. (Object Management Group, Inc., 2013)

The connecting objects consist of sequence, message flows and associations. Sequence flows dictate what order and direction the process goes in the diagram. Message flows are used to indicate what kind of informal messages cross boundaries such as different pool-lanes. An association is used for connecting references to objects belonging to the artifact category. (Object Management Group, Inc., 2013)

The swim-lane category simply holds pools and lanes, where one pool contains either one or several lanes. This is generally used for modelling interactions between different participating processes. (Object Management Group, Inc., 2013)

Data objects, groups and annotations belong to the artifacts which are mainly used for documentation and descriptions of respective modelled processes. (Object Management Group, Inc., 2013)

3.1.2 Decision Model and Notation (DMN)

This notation standard is similar to the BPMN in the sense of trying to make a notation understandable between two parties. DMN is an attempt to form an interchange format for decisions based on rulesets. A ruleset can be defined as a decision table with different hit policies. (Object Management Group, Inc., 2019)

Among the hit policies there are:

- Unique
- Any
- Priority
- First
- Collect
- Output order
- Rule order

The unique policy enforces that only one single rule can be matched. Similar to the unique policy there also is the first policy. It will not enforce a single result, but it will return the first one it can match in the table from top-down order. The any policy allows matching of several rules but only if they state the same output. The policy of priority allows several rules to be seen as triggered, but it only returns the one with the highest output priority as a match. A collect policy allows you to aggregate matched rules by utilising an operator of choice. The operator can be: number, maximum, minimum or sum. The number operator returns the amount of outputs the matching rules return. Sum, maximum and minimum do exactly what their names imply to the matching rule outputs. The collect policy can also function in a table, returning multiple results. Rule order returns a list of matching outputs, sorted by sequence. And lastly, the output order policy which takes all matched outputs sorted by decreasing priority. (Object Management Group, Inc., 2019)

Season decision

season

U	Input +	Output +	
	Weather in Celsius	season	
	integer	string	Annotation
1	>30	"Summer"	
2	<10	"Winter"	
3	[10..30]	"Spring"	
+	-		

Figure 3: Example of decision table from BPMN.io. (Camunda Services GmbH and contributors, 2020)

DMN also includes the expression language called FEEL (Friendly Enough Expression Language) in order to have a way of expressing a rule in the decision tables. In addition to the graphical view of the decision tables, you can interconnect rule-flows with their respective inputs in the decision requirement diagram (DRD). (Object Management Group, Inc., 2019)

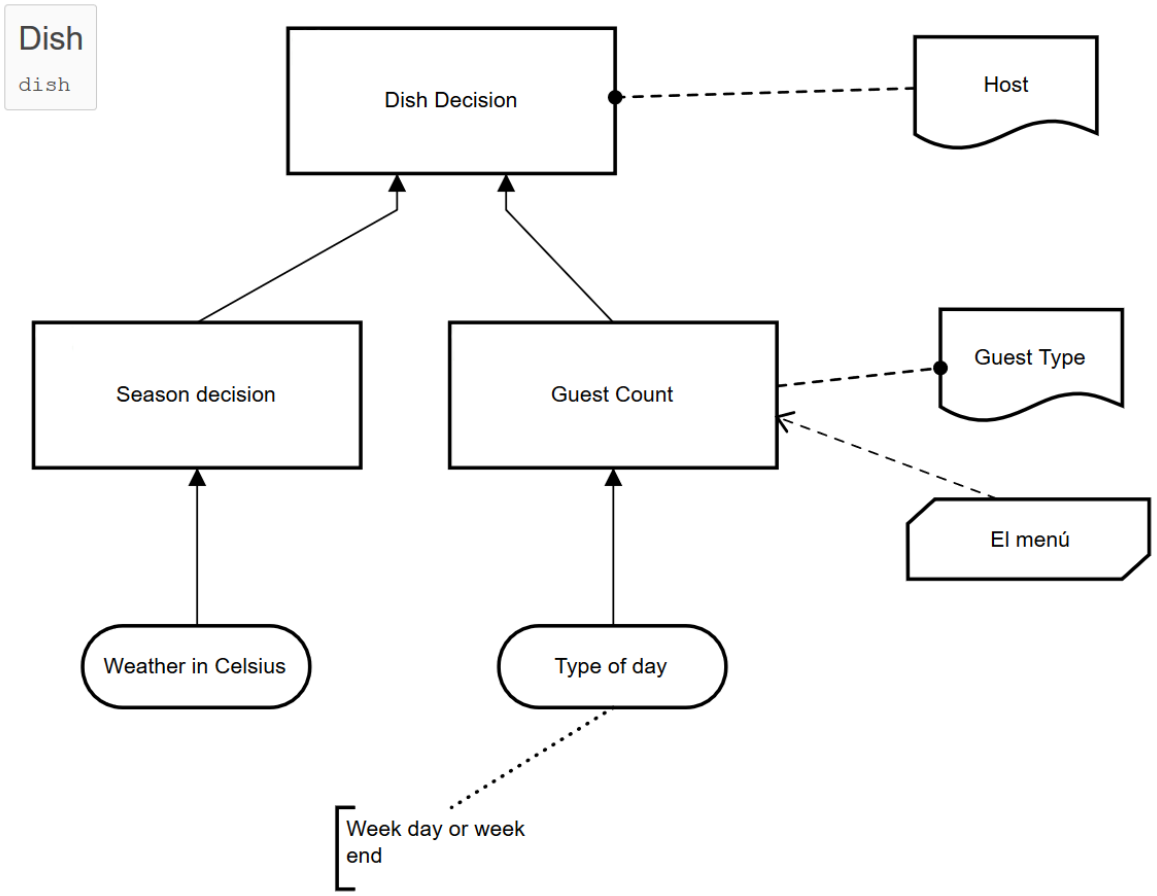

Figure 4: The decision requirement diagram (DRD) view from the same example as Figure 3. (Camunda Services GmbH and contributors, 2020)

3.1.3 Case Management Model and Notation (CMMN)

The CMMN standard is intended to be used in conjunction with the BPMN standard in order to address a broader spectrum of work methods within a given case or project which might be less clear or straight forward to implement as a process. In other words, it is a means of describing alternate cases where the process would end up being very different from time to time depending on the factors being considered. Thereby it follows a declarative design paradigm for describing the process. In practice this means that you do not need to model firm sequence flows but rather a loosely tied set of constraints that need to be fulfilled in order to progress.

A case model can be split up into stages or plans with actions which optionally have either entry or exit conditions, so-called criteria. The actions can be of different types, for instance, human-, process- or case-tasks. These actions can be either blocking or non-blocking, indicating whether the start of an activity or the end should indicate its

completion. This enables modelling of asynchronous task behaviour. Activities can also be marked as repeatable or as optional. In addition, there are event listeners like in BPMN. A certain set of events or tasks can achieve a milestone when the predefined criteria are fulfilled. (Object Management Group, Inc., 2016)

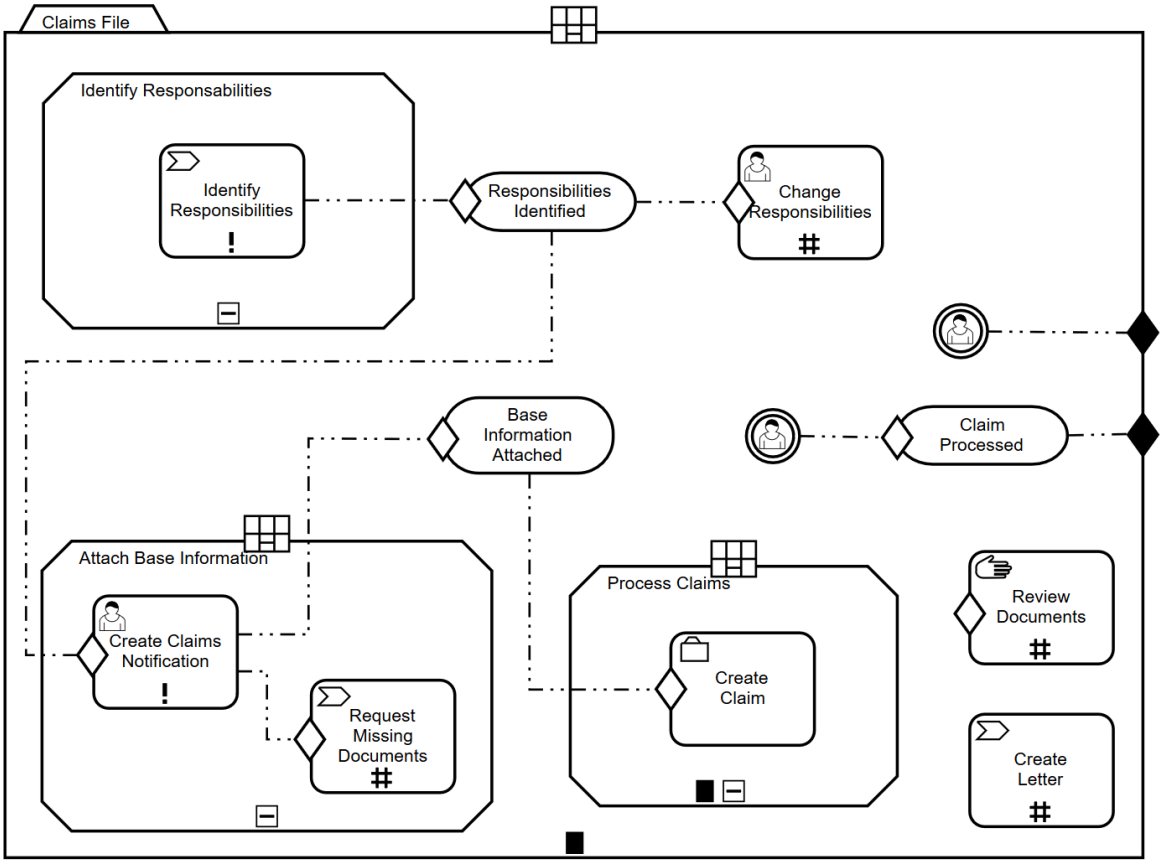

Figure 5: Another example from BPMN.io of a CMMN diagram. (Camunda Services GmbH and contributors, 2020)

3.1.4 Predictive Model Markup Language (PMML)

PMML is a predictive model description format which is meant to enable sharing of predictive models between different tools and application for analytic purposes. These shared models can be the results of common machine learning methods as well as the results of data mining (Data Mining Group, 2020).

3.2 Software architectural concepts

Different systems and ways to configure and interconnect these will be described later on. Therefore, this chapter will briefly touch on some software architectural concepts in the sections below.

3.2.1 Distributed microservices

The concept of microservices is the notion of splitting up your monolith applications into a smaller subset of services which together form the same feature set as your previous application. By dividing the responsibilities of the applications into smaller parts, it becomes easier to involve new people into the development process and to do smaller incremental updates regularly. At the same time as you gain the possibility to scale up individual areas of responsibility within the application, you can do so without duplicating the resources in order to scale a single, larger application. This enables an immutable approach of service deployments which go hand in hand with cloud native applications. By separating the configuration from the actual runtime setup, you gain deployment flexibility. By having smaller services, it is also easier to dedicate developers to a specific domain of the application, which helps if you aim to achieve practices of agile development. (Adamski, 2018)

3.2.2 REST

REST is not a standard but more of a predefined ruleset one could follow when designing restful application interfaces (APIs) in order to form machine to machine communication, server to client communication, or vice versa, over http. REST stands for representational state transfer and was coined by Roy Fielding who was one of the contributors to the HTTP specification. In general, it describes that the communication should be stateless. In other words, the receiver should always be given the contextual data, so a valid response can be returned without the need to keep track of additional context between requests. The resource endpoints of a REST-API, an application interface consisting of different URLs, should be uniform in such a way that the different methods are defined by the already existing HTTP verbs such as GET, POST, PUT and DELETE. The data presentation can be in various formats and should be indicated in the HTTP headers of the resource request along with an indicator of the ability of the request to be cached at the client side. The

formats being usually utilised are either JSON (JavaScript Object Notation) or XML (Extensible Markup Language). (Richardson & Ruby, 2007)

3.3 Development and deployment tools

To aid ongoing software development, there are many tools one can utilise. Most certainly you will at least need a way of managing versioning and enabling collaborative development if there are more than one person working on the same code base. This section touches briefly on some of these tools.

3.3.1 Git

Git is an open source project initially created when a version control system was needed in the Linux Kernel community. A version control system is a way of keeping track of file changes in a project. By having a centralized git repository, you can have multiple contributors working on the same project simultaneously with full history of changes. Git allows you to have several branches which can be worked on in parallel and then later be merged into the master branch or another sub-branch. Git is also distributed which means that for each contributor who uses the repository you have a backup since it mirrors the entirety of the given branch in the central repository in case it goes down. Git is only one of many solutions for version control, but it remains as one among the popular ones being used within the industry as of today. (Chacon & Straub, 2019)

3.3.2 Apache Maven

Maven is mainly a build tool with capability for handling dependencies for Java projects. It enables build time automation, for example running unit tests and then deploying to different targets. Its capabilities can be extended with Maven plugins, and thus it can also support projects written in other languages (Apache Software Foundation, 2019).

3.4 Runtime stack

Most of the already existing attempts to create BPM solutions are crafted in Java. The following sections will briefly cover details specifically about the runtime and its underlying features and specifications.

3.4.1 Java runtime

Java is an object-oriented programming language like Microsoft C#. It was mainly inspired by C++ but has partially taken away the responsibility of managing memory allocation from the developer due to implemented automatic garbage collection. Thus, it can be categorised as a higher-level programming language. One of the strong points of Java is that it has a versatile runtime which can be utilised on plenty of platforms. This is done by the implementation of a Java virtual machine (JVM) which takes Java bytecode and translates it to proper machine instructions (Lindholm, et al., 2018) (Evans, 2015).

There are many different JVM implementations. Owing to that, it is merely a standard that dictates what the JVM should be capable of. This allows developers to choose their approach themselves as it comes to how their VM interacts with the lower-level instructions to the respective underlying hardware. There are many different existing implementations and even programming languages which follow the same standard to target the JVM as runtime such as Kotlin, Scala, Clojure etc. (Evans, 2015)

3.4.2 Java EE

Java EE can be described as a standard consisting of a collection of specifications which conform to enterprise usage of applications written in Java. Therefore, when utilising a given Java EE product, it is often in a form of a Java application server that can run Java apps with API-interfaces and runtime conventions for having interservice dependency relations with supported methods and protocols of messaging, transactions and persistence handling coupled with dependency injection containers and Java component encapsulation. The aim with this can be seen as to provide a complete runtime environment for Java applications with their respective needs, to allow the developer to concentrate on the business logic (instead of on the surrounding implementation requirements) of said applications. (Adamski, 2018)

There are a handful of different application servers which will be mentioned later. One of these is Tomcat which is a project under the Apache Software Foundation. Another alternative implementation of a Java application server is WildFly, formerly known as the JBoss Application Server. This open source project is the foundation for Red Hat's JBoss Enterprise Application Server (EAP) offering.

3.5 Deployment stack

The deployment pipeline takes into account how a certain set of applications are being handled from the stage of being built from the version control repository to the stage of being configured and deployed into an existing environment with underlying server hardware. Therefore, the following subsections are about some of the tools being used today in cloud-based solutions.

This pipeline flow is often described as continuous integration or continuous deployment or continuous delivery (CD). Although these terms are often used interchangeably, they have slightly different meaning. With continuous integration you have automatic builds triggered by changes in your repository of choice, with proper test coverage to catch issues early, whereas with continuous delivery or deployment it generally refers to the whole process of continuous integration to the end where the actual deployment of a build or a release occurs. (Ott, et al., 2016) (Arundel & Domingus, 2019)

3.5.1 Docker

Docker is a way of handling containerisation of the entire application runtime into an image which is easier to build, maintain and deploy under different contexts without the struggles of managing virtual machines (VMs) with their respective resource needs and overhead of configuration and size. In other words, it allows developers to easily describe an image containing the underlying operating system with the instructions of setting up the needed dependencies for runtime. To make development faster, those images can be shared and extended on so called Docker repositories. (Gupta, 2016)

Each docker image can therefore be thought of as a collection of reusable layers. Internally this is due to the usage of a union filesystem. By utilising it, docker can take each description from the docker-file and branch out file changes to separate file systems for each instruction while still retaining the ability to overlay them collectively as one single file system. (Gupta, 2016)

This effectively means that if all running containers use the same base image, only the overlying layers need to be downloaded in order to do the complete automated deployments. It also speeds up the general process of building containers when the underlying layers remain unchanged and can therefore be skipped. (Gupta, 2016)

This also enables flexible multi-stage builds, where a build can account for the different build-time prerequisites but then produces a slimmed down runtime image without the now redundant build dependencies.

Due to the format of containers, all file-system changes occurring within the running container get wiped on restart. The application state has either to be saved outside the container by using external databases, or it has to be configured in a shared storage volume to which the containers can write persisted data. (Gupta, 2016)

These readily built Docker containers can then quickly be started or instantiated according to the need for the packaged application. By running multiple instances of these containers behind a reverse-proxy, one can create scalable solutions which will effectively utilise the underlying hardware in order to serve the growing demand. When it comes to interconnecting, monitoring and managing these clustered environments, the need of orchestration of such instances becomes apparent.

3.5.2 Kubernetes

Kubernetes is an open source project with its roots originating from Google, but it is now under the umbrella of the Cloud Native Computing Foundation. Its aim is to be a tool for orchestrating automation of deployments, scaling regardless of hosts or what the containerised operation is. Although this is not an end-to-end solution from the hardware level, it has proven itself through its wide adoption within the field due to more and more corporations adapting to a cloud native infrastructure. (Burns, et al., 2016) (Arundel & Domingus, 2019)

Kubernetes contains a few concepts along with so called objects (The Linux Foundation, 2019). Pods represent defined processes which need to be run in the cluster. These processes can be either a single container or several. By wrapping for example, a docker container in a pod, it gives you the ability to dictate what resources you allocate, e.g., storage, networking amongst other things. When scaling this containerised application, one can start several pods of the same container which is referred to as replication in Kubernetes. These replications are often grouped together and managed by another concept of Kubernetes controllers. (The Linux Foundation, 2019)

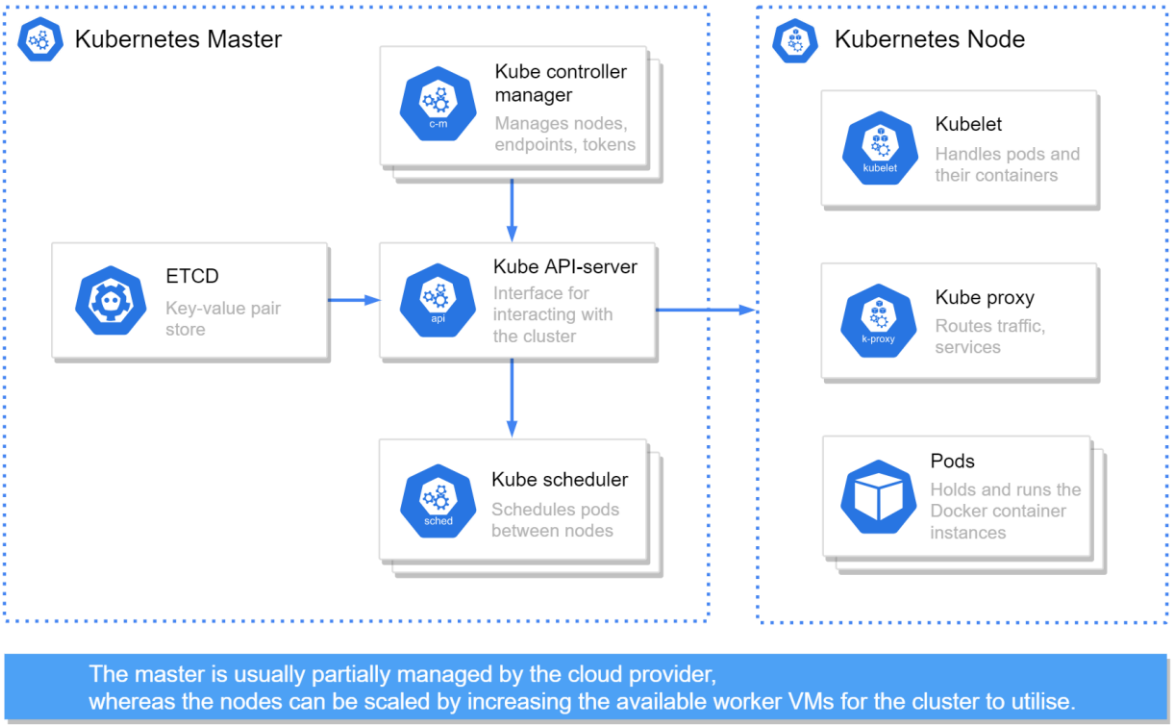

Figure 6: Illustration of how the overlying concepts tie in together to form an orchestration system for a containerised application cluster.

A controller is responsible for directing the cluster into a desired state described in configuration. This can for example be a job which would spawn a set of pods for a scheduled time. Or in other cases driving the cluster state to the desired one when pods fail or crash in unforeseen events. Due to the controller being an abstraction, there are several underlying implementations other than a job that are available for use such as daemon-set, replica-set etc. (The Linux Foundation, 2019)

Created pods are often run under concept objects called nodes. In reality these nodes can be the underlying physical host servers or virtual machines responsible for running these containers. In order for the nodes to be able to instantiate pods, each node includes its underlying services which include the container runtime, the node agent called Kubelet and the network proxy. These accompanying structural node services can vary depending on which cloud provider you happen to choose, unless you choose to maintain the Kubernetes backbone as well, where you can choose for yourself. (The Linux Foundation, 2019)

To expose a set of pods whose functionality is responsible for a coherent part of a bigger application (in accordance with the microservice architecture), those could be abstracted under a Kubernetes service. This exposes for example a way of defining reverse proxy for

scaled backends where a statically served frontend could be pointed. This is facilitated at configuration time by each pod having its own assigned dns-name and ip-address within the cluster. (The Linux Foundation, 2019)

To handle persistent data, Kubernetes introduces the concept of volumes which pods can utilise. This gives a means of storing data elsewhere than in the container itself, which otherwise would start with a clean slate each time a pod would be restarted. (The Linux Foundation, 2019)

For clusters that are getting large in size one can utilise namespaces. A namespace is a way of dividing parts of the cluster into scopes. This can be useful for example when managing resources between several developers in a larger cluster. (The Linux Foundation, 2019)

4 Solutions considered during assessment

There is a plethora of available solutions for BPM and BPA oriented systems . Most of these can be ruled out just by the licensing being based on transactions and usage in general and the core system being proprietary. The main two alternatives remaining are alternatives derived from the Drools and jBPM projects and Camunda which all happen to be open source with a commercial enterprise support tier.

4.1 Red Hat Process Automation Manager

Process Automation Manager is the Red Hat supported enterprise solution for creating and managing business rules and models by means of the BPMN and DMN. The foundation of this offering is built upon the solutions under the umbrella project KIE (Knowledge is Everything). The underlying projects are Drools, jBPM and Optaplanner amongst others. Therefore, the KIE-server (a.k.a. Process Server) and the Business Central, which are two key components of the solution, remain quite the same although being rebranded in the Red Hat offering. (Red Hat, Inc., 2020) (Red Hat, Inc. and contributors , 2020)

Drools is an open source business rule management system (BRMS) solution which is trying to tackle the problem of business process management (BPM) through several standards coupled with a supported declarative programming paradigm for developers to

use in the given rule engine. This was first done by the introduction of the Drools Rule Language (DRL) but also later on accompanied by the way of the DMN standard. (Bali, 2013)

With the rule engine covered, the jBPM project comes into the picture in order to prove a workflow engine, now based upon the BPMN standard. Optaplanner (also called Business Optimizer) is a constraint solver which can be used for optimisation problems. The line between all these projects is blurred since they refer to each other with parts and sub-projects which have been merged under different names at several occasions. Therefore, the sections below will focus on the process automation manager and its Business Central and process server (KIE-server) and their underlying runtimes. (Red Hat, Inc., 2020)

4.1.1 KIE-Server

The KIE-server is the backend responsible for creating instances running the rule models and artifacts (built java projects) in such a way that they become available as discoverable rest services ready for use. In this way the rules become available for use within the BPMS solution and potentially outside, if needed. (Red Hat, Inc., 2020)

4.1.2 Business Central

Business Central is a service which provides a frontend and a backend for managing and creating model descriptions. In other words, it is a platform for managing the lifecycle for rule and process artifacts which need to be deployed to a KIE-server container in order to be run. Depending on user permissions, this can become a control panel for relevant areas and a tool for others. (Red Hat, Inc., 2020)

4.1.3 Red Hat Enterprise Application Platform (EAP)

The enterprise application platform is an alternative for deploying enterprise grade Java EE applications. Among the provided features are secure clustered deployments suitable for containerisation in OpenShift. In comparison to the common Drools setup with WildFly or JBoss Web Server which relies on Apache Tomcat as a Java Application Server, EAP is a supported platform based upon WildFly. (Red Hat, Inc., 2020)

4.1.4 OpenShift

OpenShift is an open source project maintained by Red Hat with an attempt at offering an end-to-end platform for hosting managed Kubernetes clusters with their respective applications. OpenShift itself is an abstraction of Kubernetes with additional tooling. The main difference between OpenShift and Kubernetes is that the former is a complete product offering, whereas the latter is just an open source project. This gives customers the option of getting support regarding hosting their containers with OpenShift. In addition, Red Hat offers OpenShift on already existing cloud providers like Amazon Web Services (AWS), Google Cloud Platform (GCP) and Microsoft Azure. (Adamski, 2018) (Red Hat, Inc., 2020)

4.2 Camunda

Camunda is another offering that implements the BPMN and DMN standards through a collection of services.

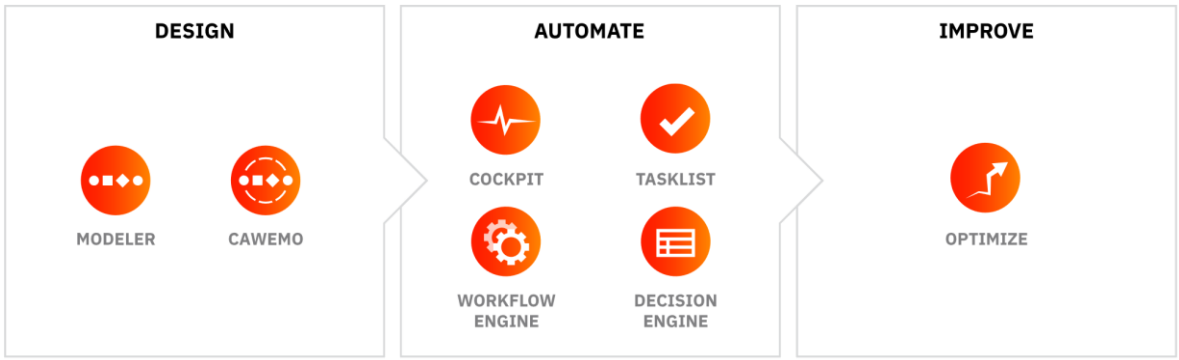

Figure 7: Overview illustration of Camunda BPM products. (Camunda Services GmbH, 2020)

The Camunda Modeler is a standalone application which utilises the open source projects Camunda has under the bpmn.io group. The modeler itself is a tool which implements these web technologies in order to form an application for editing the diagrams of the BPMN, DMN and CMMN standards. In addition to this it is also extensible and lets you do direct deployment of your process diagrams or decision tables into an existing Camunda installation, given the correct authorization. (Camunda Services GmbH and contributors, 2020)

Cawemo is a stripped-down version of the Modeler which utilises the same open source projects to enable collaborative drafting and development of diagrams with versioning and commenting as an in-browser experience. This tool is available for free with a registered account, or you could deploy it on your own premises. The whole code base of the Camunda services is available as open source, except for functionality which the licensed enterprise supported versions include (which Cawemo essentially is, for on-premise deployed usage). (Camunda Services GmbH, 2020)

Camunda Cockpit is a web frontend (for the workflow engine) which provides a means of monitoring, analysing and resolving incidents (issues caused by process errors) in deployed processes. Furthermore, it provides an insight into the history of previous processes as well. (Camunda Services GmbH, 2020)

Tasklist is a similar frontend which focuses on offering the end-user experience for resolving tasks where persons need to be involved in a described BPMN process. (Camunda Services GmbH, 2020)

The workflow engine is the backbone of Camunda. It keeps track of deployed processes and active process-instances, and it orchestrates the given modelled business processes. It can either be deployed as a standalone service or embedded in your own Java application. (Camunda Services GmbH, 2020)

The decision engine is responsible for handling the execution of deployed DMN tables or diagrams. It is embedded by default in the workflow engine but can be utilised as a standalone application and embedded in your own Java applications likewise. (Camunda Services GmbH, 2020)

Camunda Optimize is a web-app you get with the licensed enterprise version. Optimize gives the feature set for creating analytical reports, dashboards and visualisation of past and running processes. BPMN diagrams can for example be presented with heatmaps for comparison of different time periods needed for different tasks and such. This enables you to find bottlenecks in the process-flow quite easily. For process targets you can also set alerts for certain events. (Camunda Services GmbH, 2020)

5 Evaluation

First thing being considered is how we can apply the solution beside our existing services and then how we will be able to scale it according to our usage when we have future proofing in mind. The solution itself should be grounded on robust and proven methods while it offers reusable components for usage within the solution as well as the possibility for external systems to utilise it. The platform should in other words offer the functionality to deliver value for other systems at the same time as it should be able to harness others. Therefore, the design should be application interface (API) centred in such a way that it can consume and provide in an extensible manner. By having the core of the process driven by data, it should also have extensive logging capabilities for the ability to trace changes for proper auditing. This is a major corner stone of the solution if it is to be implemented to deliver continuous automation between different enterprise resource planning (ERP) systems as well as being the core system to initiate changes to revisory master data with accompanying approval workflows. The idea behind having a citizen driven development community is that each department should have the possibility to utilise suitable tools to implement their respective processes in such a manner that they do not need the expertise of a full-stack developer. However, the possibility to implement more intricate functionality should still exist to a certain extent. As such user friendliness accompanying with a valuable technical platform for automation needs to be achieved without hard vendor lock-ins with the added benefit of the ability to be developed in-house.

The idea behind the chosen solution would be that it first would get a proof of concept use case where it would be able to demonstrate some of its capabilities. After a demonstration with internal stakeholders it would hopefully get a greenlight for a real pilot project with practical use.

5.1 Red Hat Process Automation Manager

The process automation manager is a project which is based on open source projects which have been existing in the BPM space for quite some time. During the years a lot of changes have taken place. At first sight Red Hat Process Automation Manager has comprehensive documentation available and the underlying projects have a reoccurring

release schedule with one to two releases each month. The available artifacts and examples suggest utilising WildFly or its derivatives for hosting it as a Java EE application.

Since the process automation manager consists of several components, simply speaking of the business-central & KIE-server (with their respective sub components), they can be deployed either under the same Java EE server or under separate instances. The configuration of the underlying WildFly server can also be managed differently depending on whether you choose to run your instances in standalone or domain mode. The capabilities of these modes are the same since it is how the configuration is handled where it differs. Since many of the responsibilities fall back on the underlying Java application server, it has some inheriting importance of being configured correctly in order to support the running applications. (Fugaro, 2015) (Red Hat, Inc., 2020)

In domain mode configuration you set up a master domain controller which then will be able to hand out the deployed applications and configurations selectively to a set of defined slave nodes. Despite the possibility to make this work in a container environment in the cloud, it goes against the principle that each container answers to its very own responsibility in a microservice architecture. Nonetheless, it is useful knowledge if you are to set up a simpler development environment for extensions to Business Central or for service tasks for local development. (Fugaro, 2015)

Testing was done to a certain extent relating to how this configuration could be done with WildFly or with Red Hat's EAP offering. This was done through utilising a virtual machine instance in Google Cloud on Debian Linux with some scripting to handle the application server configuration with the possibility to refer to different versions and setups (e.g. domain or standalone mode) in order to compare these. There were quite a lot of naming changes between the projects which can cause some confusion. Another interesting fact is that the readily available releases of the drools and setups seemed to refer to WildFly 14 since there were four newer major revisions since that version. Red Hat EAP follows a completely different release cycle with unmatching increments.

Business Central complicates things also in regard to how it handles the internal projects and deployments you create or add to the platform. Projects can either be imported or created solely within the provided Business Central web-based toolset. These projects are then stored in an internal virtual file system (VFS) based repository (in this case a git

repository). Imported repositories can also be kept as external by configuring git hooks to order Business Central to keep in sync. (Red Hat, Inc., 2020)

Each project, either created or imported, is in practice a Maven project. This means that in practice Business Central is an abstraction to prove an integrated development environment (IDE) around complete Java Maven projects. It handles creating, managing and deploying these projects by saving the built artifacts (the build result of Maven projects) onto a Maven repository which can either be local to Business Central or external (such as an external Nexus repository). The resulting Maven artifacts are then to be used in a KIE-server deployment. (Red Hat, Inc., 2020)

With the task of providing web-based tooling on top of Maven projects for definitions and serving it as a low-code experience according to the standards, it becomes hard to keep it from seeping some of its complexity onto the design time for the end user. One apparent example is how the editor locks the entire project specifically to you in order to hinder others from working on it at the same time due to all changes being committed on save to the underlying git repository. Although the possibility to do work on separate branches and pull requests (which are named change requests within Business Central) still exists, it might add to the learning curve of the end-user.

Due to the inherited complexity of abstracting the tools of the whole development process into one single platform, it becomes troublesome when it comes to scaling Business Central itself. The approach would be aimed at scaling vertically (by giving the machine running the application better hardware resources) since this is preferable from the configuration standpoint. Otherwise one would have to set up a solution for file inter-locking mechanisms to keep several Business Central instances in sync with the same contents in the internal VFS store if you so choose to run the central in high availability as the KIE-servers are meant to be run in. In addition, a message broker between the instances would need to be setup. Although this is still technically possible, it is something that is said to be in technology preview and the existing documentation gives different hints depending on what source you choose to read from. This can also be seen as unnecessary from a domain driven design architecture standpoint. Especially if it is to be used solely during development time. But if that is the case, there might actually be a need

for a separate frontend for letting the end-users start and interact with processes which in turn put the attention on the KIE-servers. (Red Hat, Inc., 2020)

From that standpoint you will have to build your own frontend while relying on the APIs of the KIE-servers to interact with the processes or rule definitions. This can become tricky when you have different deployments on different KIE-server instances since all results from each instance would need to be aggregated to get a holistic view of all deployed processes. Luckily, this can be solved by relying on the smart routing solution from Red Hat for this use case. It enables you to communicate with the cluster of KIE-servers in a similar fashion like having them behind a reverse proxy albeit a bit differently. This router enables you to interact with all existing KIE-instances as one single KIE-server. (Red Hat, Inc., 2020)

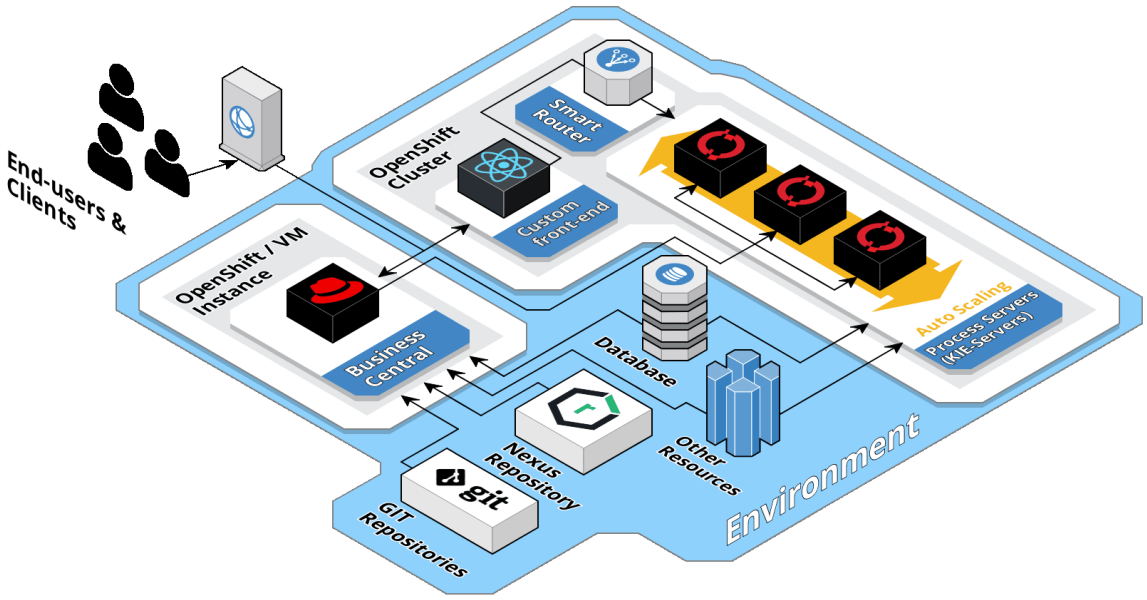

Figure 8: Visualisation of what a potential hybrid cloud setup of the Red Hat Process Automation Manager could look like.

When it comes to the management of these KIE-server instances, there are several different approaches. The set of KIE-servers running in parallel can either be run in managed or unmanaged mode. This affects how the Business Central projects are handed over to the KIE-servers. Usually the KIE-servers are run with a head controller. In a simple setup this controller usually resides in Business Central itself. But when it comes to multi-stage environments (such as development, quality assurance and production environments), it is not unusual that the KIE-controller sits on the cluster and not on the

Business Central instance. When running the KIE-cluster in managed mode with the separate controller, it can be configured to poll for new artifacts on the Maven repository and then automatically deploy them. When running the unmanaged mode, you will be able to achieve greater immutability with the releases which means that a rebuild and a redeployment occur each time a new process is deployed. This approach follows the microservice architecture closer than the managed approach. Whether you want to do this depends on what you prefer and how you want to approach the cluster management. (Red Hat, Inc., 2020)

The KIE-servers can, like Camunda, be embedded in a Spring Boot application as well, but the benefit here is maybe not as big as with Camunda. Since Red Hat provides support for running the KIE-servers in unmanaged mode with OpenShift, which has source to image tools, this could actually be the less maintenance demanding approach of the two. However, if there is an existing cluster, the embedded or the other approaches are still equally valid, with varying extents of support. (Red Hat, Inc., 2020) (Adamski, 2018)

When it comes to providing integrations to external services, you can implement custom work-item handlers in Java. The characteristics of each work handler can be described in a format which allows Business Central to provide them as custom BPMN tasks in the model editor. These work handlers can be in the form of a Maven project whose artifacts are added to the Business Central repository. From there you can either enable the handler in the administrator panel to allow all projects to install them in the settings menu, or you can add them manually to each project by adding the artifact as a dependency with the correct work-item definitions. (Red Hat, Inc., 2020) (Maio, et al., 2014)

5.2 Camunda

Camunda offers a solution which is concentrated on one backend which can either be run standalone in parallel connected into the same database or embedded into your own Java EE application. This gives you the possibility to choose how far you as a developer want to go with developing around the given solution yourself.

A plus is that Camunda offers standalone tools for drafting and then actually developing and deploying processes into the workflow engine. Cawemo is a web platform which can be deployed internally and can be configured to sync the already deployed processes for

drafting on reiterations to improve the workflow. The Camunda modeler is an implementation of a collection of open source projects under the bpmn.io namespace, which can be reused in custom-made frontends or in your own modelling tools, if so needed. (Camunda Services GmbH and contributors, 2020)

With the Camunda modeler you can refer to service tasks in the BPMN workflow in which you either can do some light scripting or actually refer to a java class path for an implemented java delegate to handle the action needed. This is becoming quite handy since developers could build a reusable toolset to manage all the interactions between different systems needed as such. (Camunda Services GmbH and contributors, 2020)

The service task libraries would also need to be deployed in conjunction either with the specific process in mind or with Camunda itself to become reusable across a set of processes. This is something that need to be considered when one starts thinking about offering it as a service. Should the end-users be able to deploy their accompanying delegates and service functions as complete Java web applications, or should it be constrained to a pre-existing tool set available for all, which is leaner. (Camunda Services GmbH, 2020)

Another option for the service tasks is that they can be separated entirely from the workflow engine runtime to become decoupled as external clients. The communication between the service client and the engine can happen in different ways in push or pull configurations over REST or equivalent APIs. The disadvantage with this is that it becomes harder to do unit testing. Furthermore, it can also be detrimental from the transactional standpoint albeit you gain the possibility to create services in your choice of language. This approach, however, indirectly opposes an open service landscape if you start building a separate application tailored specifically to Camunda instead of being targeted for generic availability. The generic API for said external service could instead be wrapped as an abstraction provided by a Java delegate to the workflow engine. In that way external components gain value outside a single domain of usage which would be preferable. (Camunda Services GmbH, 2020)

For an approach concentrated on the tooling workflow, which Cawemo in combination with the Modeler provide, it is not that bad of an approach to run Camunda in embedded

form in conjunction with for example the Spring Framework. (Camunda Services GmbH, 2020)

The needed toolbox of service task delegates, to incorporate external services, can be concentrated into one main project where the whole workflow-engine lives. That simplifies the management of deployment of the core addons. The additional benefit of using an embedded engine is that you can easily build a proper unit test suite to validate whether the pluggable services work as expected.

To handle security concerning the provided web tools (cockpit, tasklist), one can rely on the Spring Security framework to provide a single sign-on (SSO) experience and authorization around the rest-API. To get users and user groups, one can incorporate it with LDAP to provide user authorization. One reoccurring theme among the community projects which provide LDAP integration or Keycloak integration (which is another Red Hat offering) is that they seem to be lacking support for multi-tenacy configurations. (Karanam, 2018)

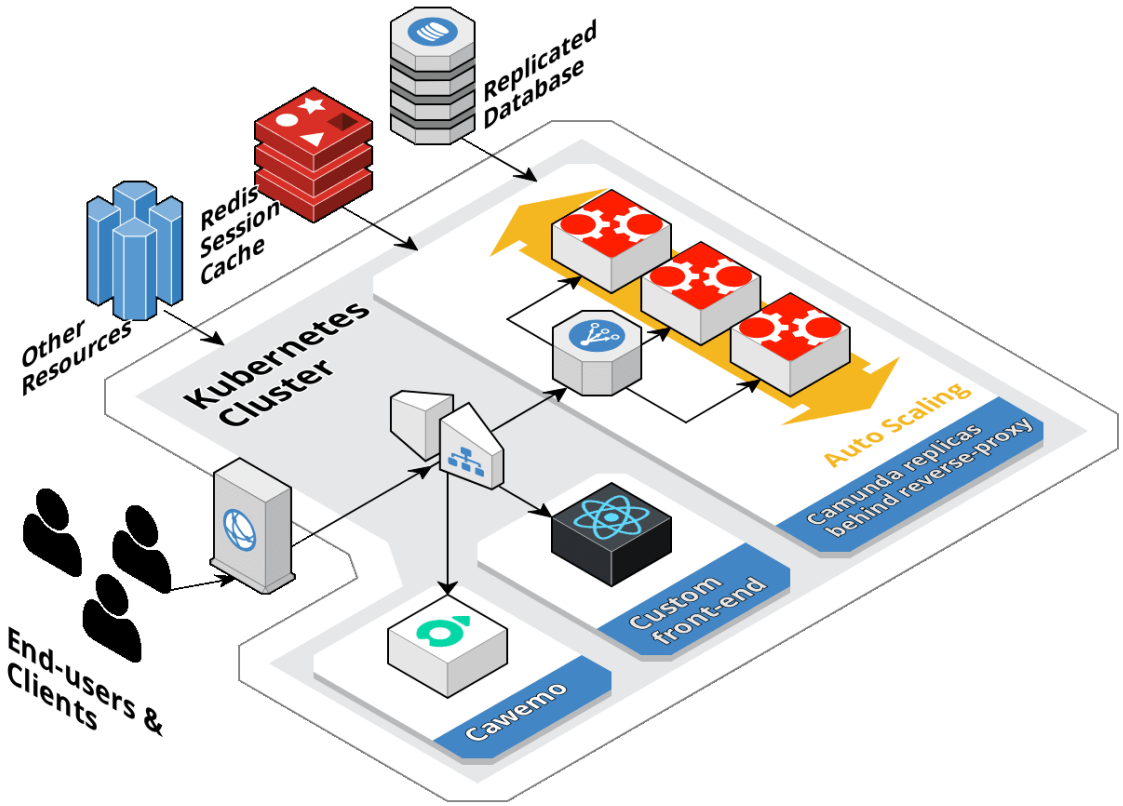

Figure 9: Theoretical cluster environment configuration for running Camunda.

By having a single project being responsible for the core of the backend, it becomes quite straight forward to include builds for readily available docker containers which then can be put in different test runtime environments with the abstractions of Kubernetes and surrounding release pipelines to get there.

Session handling for the replicated embedded Camunda instances can be handled by the session handling support which the Spring framework already provides. In this case that could be a Redis instance which is usually used for varying caching applications with structured data or where some sort of message broker is needed. (Karanam, 2018)

This would mean that each time the user request does not end up at the same replicated instance, after going through the front-facing reverse-proxy and the replica instances have been scaled down, the instance itself would check in with the Redis cache to check for a valid previously existing session. Using that session instead allows the user to continue without getting prompted to authenticate again.

5.3 Conclusion

Camunda has promising and extensible modelling tools available. If the current modelling tools do not have a needed feature set, you can either extend the existing Camunda Modeler or roll your own implementation using the open source libraries. Since the entire modeler is built on web-technologies, you can easily embed it in your own applications. These could be in your own frontend served as a website or as a desktop application wrapped in Electron.js (similarly as the Camunda Modeler itself). Cawemo brings a separate platform enabling collaborative real-time model tools for drafting and documentation.

Holistically this provides a simple toolset to promote for a citizen developer community with the focus staying on the described models. This in turn simplifies the underlying infrastructure to an extent without disallowing you to go down the route of creating separate domain specific Java applications on an application server in the future.

Red Hat, on the other hand, has embraced the concept by abstracting the modelling and management tools around full Java projects. This is a large area to cover in order to make

it completely user friendly, and therefore it can feel rough around the edges when using its modelling tools.

There is also a difference in how the standards are supported in these tools. Camunda supports modelling for BPMN, DMN and CMMN, whereas Red Hat PAM only supports modelling for BPMN and DMN although support for execution of CMMN models still exists. This can be seen as a bit odd although CMMN can be claimed to be harder to grasp from the presentation view than a BPMN diagram in comparison due to its mere nature. Therefore, Red Hat seems to suggest relying just on BPMN modelling with a custom milestone activity to mimic the CMMN standard, but supposedly in a more readable manner. In contrast, Red Hat PAM supports the additional standard of PMML which can be seen as valuable but out of scope from the current consideration. (Red Hat, Inc., 2020)

The way of modelling BPMN can be different since Camunda allows you for example to join several sequence flows into one single task activity and adding boundary events on top. The process automation manager on the other hand enforces you to re-join all flows through gates before routing them to any activities. Some of these can be discussed as valid enforcement of best practices in BPMN modelling while it can be inconvenient from the perspective of small processes.

Both solutions are capable in their respective ways, but both do not have an included frontend capable enough for delivering such an end-user experience we were targeting. Camunda allows you to extend their respective frontends to a certain extent with plugins. The Tasklist can for example refer to embedded forms in deployed processes (when processes are deployed as separate projects on the Java Application server) or to external ones. That makes the process of creating forms and taking them to use more inflexible than deploying the process definitions themselves (from the viewpoint of the embedded workflow-engine shared by all). Red Hat has a more involving approach to the data parameters and object models, on a per task and process basis, throughout its modelling tools. That allows you to generate a foundation for forms in the tools available in Business Central. These forms can be used within the built-in equivalent task lists of Business Central and so on. That would introduce the whole central workflow development tools to all users with different roles and permission sets. From our viewpoint this should be a separate usage domain being concerned with the underlying foundation that Business

Central consists of in retrospect. This can be seen as an opportunity to create a custom frontend for providing an abstraction for the intricacies which the underlying systems contain. One of these could for example be the so-called business keys which can be given as contextual clues when starting a new process instance.

6 Implementation

6.1 Proof of Concept

This chapter describes the work regarding the creation of a proof of concept demonstration utilising the Camunda platform. For delivering the means for the end users (all employees within Wärtsilä) to initiate a process in Camunda, the default task-list web application might become shorthanded. For us it would mean that we would decide on creating an alternative frontend for this. The default Camunda Tasklist has some extension possibilities such as deploying custom forms for handling user tasks or referring to external ones. For us this is not quite enough. There is a need for a more streamlined way of working without having it ending up as a work effort for a developer when a new user task for a process needs to be implemented.

6.2 Use case

The proof of concept would ideally be able to achieve similar functionality as the task-list but with the ability to start new processes by filling in a web based form and then let the process continue and eventually end up at an approver's table in a similar fashion by using user tasks from within the BPMN process definition. In order to handle the forms part of the frontend, we have to respect the fact that it is not only the business process and decision models that should be doable by the end-users but also the form creation. When forms should be doable in a similar manner with low-code tools, a similar definition format needs to be found or implemented.

6.3 Implementation

The idea behind a custom task-list is that it should have a form schema or description format that easily can be modified with a web-based form builder and then easily referred to and used in the actual process. The schema itself can be serialized in a suitable format like JSON and then stored elsewhere and retrieved when needed. The definition itself could be referable from the BPMN standpoint via the start events or user task elements via the form-key property on them. A library or framework called Formio became of interest for this functionality and will be utilised for the proof of concept frontend.

In this case Camunda provides some sample projects, under Camunda Consulting, which one can use as a foundation to build further upon. I chose to go with the React.js based one since there are other services within Wärtsilä which are already dependent on it.

A recurring technical challenge among the frontend frameworks is how to manage the state in the hierarchy of web components. It is easy to fall in the pit of prop drilling where you end up passing properties top-down in a long hierarchy of react components in order to get the needed properties into the nested components you have written. Consequently, you get the accompanying problem with unnecessary re-renders and how to trigger changes to the overlying state from the composition-based component hierarchy in the DOM.

Many developers usually opt for utilising Redux in conjunction with React, but this time I opted to utilise Atlassian's react-sweet-state library which offers much of the same functionality. In comparison it lets you have a global state and then nested and separate scoped states with the same support for state selectors with the addition of having asynchronous dispatch actions without relying on the Redux-thunks library. Middleware is supported for manipulating the state on change as well, like in Redux, and it was utilised for managing the rest API calls to the Camunda backend. One drawback by utilising middleware is that the state changes are happening outside the reducer actions as a side effect. This makes it harder to debug when things go wrong. Therefore, it might help with the existing Redux debugging browser extensions, which both of the frameworks support.

In order to tackle immutability with a more complex state which has nested objects due to the REST-API responses being stored, the Immer library was utilised, which in turn uses

JavaScript ECMAScript 6 (ES6) standard for proxies under the hood to always return a new object with the reflected changes when you mutate the properties of the draft objects. Even if you can replace the default mutator in the reducer to always use Immer, so that you can always mutate the state in the reducer actions, there is an accompanying performance cost for the convenience. Therefore, it becomes a useful tool when being used as opt-in within selected reducer actions.

The reasoning behind the need for immutability is that React will not see the changes to nested sub properties due to the fact that it uses a shallow compare for props. This means that all the parent objects in the structure need to be recreated with the previous values and new values so that the path up to the root object will be seen as changed. Otherwise it will result in a stale state which then results in outdated component props being passed down, hindering needed re-renders to display changes. Another pitfall is when you refer to an immutable state object within an event handler where it might become stale as well due to JS closures when React batches state changes to hinder excessive re-rendering. To further help the state structure for usage with API data being stored, one can use the Normalizr library to split up embedded entities from the API response into separate sections according to schemas in order to store them in the state.

Since Formio is internally dependent on bootstrap for cascading style sheets (CSS), I opted to use it for the whole frontend. For the time being, the form definitions were mocked and saved as JSON files in the frontend itself. For future use this would be handled by a separate service to create and store them.

6.4 Outcome

The end result was put in a virtual machine (VM) instance in the Google Cloud Platform (GCP) together with a trial enterprise instance of Camunda serving as the backend. The proof of concept application was presented several times for different stakeholders. It was approved to receive funding for a pilot project for practical use already on the first showcase.

Figure 10: A pair of views taken from the resulting proof of concept frontend.

7 Conclusion

This has undoubtedly been a big learning journey and opportunity to dive into full-stack development. Besides the technology the topic is quite a wide area to cover. Therefore, there is much which is left out of this work. The actual decision, on solution of choice, will in the end likely fall back on the respective product licensing terms.

The whole topic around business process automation (BPA), or rather business process management systems (BPMS), is an area which I could see to become more commonplace than it is today. For companies already looking into or having already established the use of robotic process automation (RPA), this could be the next natural step forward for getting a backbone system for all process automation and integration needs. The BPMN standard will definitely be part of that and, as a matter of fact, it is already part of it to a growing extent. The concept is not new. It has existed for a long time and might be picking up more traction in the near future, which will be interesting to follow.

Those who are interested in taking a look at the Red Hat solutions should definitely spare some time to do so bearing in mind the different sources and amount of documentation that comes with the surrounding stack, which you can opt for. In the offering Red Hat provides there are also more products which might become of interest besides the BPMS oriented solutions which could be a future investigation. They do also have a separate

product upcoming called Kogito which is aimed at being a cloud native alternative with the possibility of being compiled to native code for fast containerised performance. Since it is in active development with no stable builds, it was not considered during the time of writing. It could, however, become an interesting option worthy of consideration in the future.

Regarding the frontend implementation, it could be a beneficial point in the future to investigate in using TypeScript instead of plain JavaScript to avoid unnecessary errors due to missing type safety. This inspection could delve deeper into how Typescript could be utilised in conjunction with React.JS and the state handling library of choice.

In regard to Java build systems, I had interesting finds when attempting to optimise build-time spent in docker containers. Maven usually downloads and caches copies of all needed libraries listed as dependencies. This is done each time if you start your Maven builds with one single Maven install command in the docker container. This can considerably slow down the build jobs with larger projects. One way to remedy this problem is to copy the Maven pom file containing the project descriptions separately from the source code first into the docker image and then attempt to run the Maven commands for preparing the project for offline mode. This would in theory allow you to create a separate docker layer which would automatically be skipped as long as the pom file does not change and thus skip downloading dependencies all over. This was fine until I started experimenting with packaging the build artifacts in different ways through Maven plugins after the build-time completion. Then the issue of Maven not preparing the dependencies for said Maven plugins for offline mode was found even though it should. One way around this was to run the Maven install command straight on the pom file without the source code. But this approach not only interferes with any post build activities but also is reliant on letting it fail and is therefore more of a hack than an actual solution. Hence, I found it might be of interest to investigate how other build tools are in comparison, such as Gradle.

Finally, I want to give my sincere thanks to the many people who have contributed to my thesis and who have supported me in my endeavour to reach this milestone. Many thanks to my supervisors, both from the school and work side and, furthermore, the entire team of wonderful colleagues at Wärtsilä who I have had the privileged opportunity to work with. And at last and not to be forgotten, the reviewing teachers at Novia. Thanks, all of you.

8 References

Adamski, T., 2018. *Hands-On Cloud Development with WildFly.* Birmingham: Pact Publishing Ltd..

Apache Software Foundation, 2019. *What is maven?.* [Online]
Available at: https://maven.apache.org/what-is-maven.html
[Accessed 27 December 2019].

Arundel, J. & Domingus, J., 2019. *Cloud Native DevOps with Kubernetes.* 2 ed. Sebastopol: O'Reilly Media, Inc..

Bali, M., 2013. *Drools JBoss Rules 5.5 Developers Guide.* 1 ed. Birningham B3 2PB, UK: Packt Publishing Ltd..

Burns, B. et al., 2016. Borg, Omega, and Kubernetes. *ACM Queue*, March.

Camunda Services GmbH and contributors, 2020. *BPMN.io - CMMN demo.* [Online]
Available at: https://demo.bpmn.io/cmmn
[Accessed 30 April 2020].

Camunda Services GmbH and contributors, 2020. *BPMN.io - DMN demo.* [Online]
Available at: https://demo.bpmn.io/dmn
[Accessed 30 April 2020].

Camunda Services GmbH and contributors, 2020. *Camunda Docs - Delegation Code - Java Delegate.* [Online]
Available at: https://docs.camunda.org/manual/7.12/user-guide/process-engine/delegation-code/#java-delegate
[Accessed 10 May 2020].

Camunda Services GmbH and contributors, 2020. *GitHub - camunda/camunda-modeler.* [Online]
Available at: https://github.com/camunda/camunda-modeler
[Accessed 23 February 2020].

Camunda Services GmbH, 2020. *Camunda Best Practises - Invoking Services from the Process.* [Online]
Available at: https://camunda.com/best-practices/invoking-services-from-the-process/
[Accessed 10 May 2020].

Camunda Services GmbH, 2020. *Camunda products - Camunda BPM.* [Online]
Available at: https://camunda.com/products/
[Accessed 27 May 2020].

Camunda Services GmbH, 2020. *Deciding About Your Stack.* [Online]
Available at: https://camunda.com/best-practices/deciding-about-your-stack/
[Accessed 11 May 2020].

Chacon, S. & Straub, B., 2019. *Pro Git.* s.l.:Apress.

Data Mining Group, 2020. *Data Mining Group - PMML 4.4.* [Online]
Available at: http://dmg.org/pmml/v4-4/GeneralStructure.html
[Accessed 27 March 2020].

Evans, B., 2015. *Java: The Legend.* 1 ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472.: O'Reilly Media, Inc..

Fugaro, L., 2015. *WildFly Cookbook.* Birmingham: Packt Publishing Ltd..

Gupta, A., 2016. *Docker for Java Developers.* 1 ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472.: O'Reilly Media, Inc..

Harmon, P., 2014. *Business Process Change.* 3 ed. 225 Wyman Street, Waltham, MA 02451, USA: Elsevier Inc..

Karanam, R. R., 2018. *Spring: Microservices with Spring Boot.* 1 ed. Birmingham B3 2PB, UK: Packt Publishing Ltd..

Kiefer, N., 2019. *Github - bpmn-io/vs-code-bpmn-io.* [Online]
Available at: https://github.com/bpmn-io/vs-code-bpmn-io/tree/master/resources/bpmn
[Accessed 30 April 2020].

Lindholm, T. et al., 2018. *The Java® Virtual Machine Specification - Java SE 11 Edition.* [Online]
Available at: https://docs.oracle.com/javase/specs/jvms/se11/jvms11.pdf
[Accessed 2 January 2019].

Maio, M. N. D., Salatino, M. & Aliverti, E., 2014. *jBPM6 Developer Guide.* 3 ed. Birmingham B3 2PB, UK: Packt Publishing Ltd..

Object Management Group, Inc., 2013. *Business Process Model and Notation (BPMN), Version 2.0.2.* [Online]
Available at: https://www.omg.org/spec/BPMN/2.0.2/PDF
[Accessed 13 Januari 2020].

Object Management Group, Inc., 2016. *Case Management Model and Notation (CMMN) - Version 1.1 with change bars.* [Online]
Available at: https://www.omg.org/spec/CMMN/1.1/PDF/changebar
[Accessed 25 January 2020].

Object Management Group, Inc., 2019. *Decision Model and Notation, Version 1.3.* [Online]
Available at: https://www.omg.org/spec/DMN/1.3/PDF
[Accessed 27 April 2020].

Ott, B., Pham, J. & Saker, H., 2016. *Enterprise DevOps Playbook.* 1 ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc..

Red Hat, Inc. and contributors , 2020. *KIE-Group.* [Online]
Available at: https://www.kiegroup.org/
[Accessed 11 May 2020].

Red Hat, Inc., 2020. *OpenShift - What is OpenShift?.* [Online]
Available at: https://www.openshift.com/learn/what-is-openshift
[Accessed 11 May 2020].

Red Hat, Inc., 2020. *Red Hat JBoss Enterprise Application Platform 7.3 - Introduction to JBoss EAP.* [Online]
Available at: https://access.redhat.com/documentation/en-us/red_hat_jboss_enterprise_application_platform/7.3/pdf/introduction_to_jboss_eap/Red_Hat_JBoss_Enterprise_Application_Platform-7.3-Introduction_to_JBoss_EAP-en-US.pdf
[Accessed 7 May 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager 7 Component Details.* [Online]
Available at: https://access.redhat.com/articles/3463751#RHPAM77
[Accessed 10 May 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager 7.7 - Creating Red Hat Process Automation Manager business applications with Spring Boot.* [Online]
Available at: https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.7/pdf/creating_red_hat_process_automation_manager_business_applications_with_spring_boot/Red_Hat_Process_Automation_Manager-7.7-Creating_Red_Hat_Process_Automation_Manager_
[Accessed 2 May 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager 7.7 - Custom tasks and work item handlers in Business Central.* [Online]
Available at: https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.7/pdf/custom_tasks_and_work_item_handlers_in_business_central/Red_Hat_Process_Automation_Manager-7.7-Custom_tasks_and_work_item_handlers_in_Business_Central-en-US.pdf
[Accessed 25 April 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager 7.7 - Designing your decision management architecture for Red Hat Process Automation Manager.* [Online]
Available at: https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.7/pdf/designing_your_decision_management_architecture_for_red_hat_process_automation_manager/Red_Hat_Process_Automation_Manager-7.7-Designing_your_decision_management_archi
[Accessed 17 April 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager 7.7 - Installing and configuring Red Hat Process Automation Manager in a Red Hat JBoss EAP clustered environment.* [Online]
Available at: https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.7/pdf/installing_and_configuring_red_hat_process_automation_manager_in_a_red_hat_jboss_eap_clustered_environment/Red_Hat_Process_Automation_Manager-7.7-Installing_and_confi
[Accessed 03 May 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager 7.7 - Managing and monitoring KIE Server.* [Online]
Available at: https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.7/pdf/managing_and_monitoring_kie_server/Red_Hat_Process_Automation_Manager-7.7-Managing_and_monitoring_KIE_Server-

en-US.pdf
[Accessed 10 May 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager 7.7 - Planning a Red Hat Process Automation Manager installation.* [Online]
Available at: https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.7/pdf/planning_a_red_hat_process_automation_manager_installation/Red_Hat_Process_Automation_Manager-7.7-Planning_a_Red_Hat_Process_Automation_Manager_installation-en-US.pdf
[Accessed 18 April 2020].

Red Hat, Inc., 2020. *Red Hat Process Automation Manager Supported Standards.* [Online]
Available at: https://access.redhat.com/articles/3642982
[Accessed 10 May 2020].

Richardson, L. & Ruby, S., 2007. *RESTful Web Services.* 1 ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc..

The Linux Foundation, 2019. *Kubernetes concepts.* [Online]
Available at: https://kubernetes.io/docs/concepts/
[Accessed 27 December 2019].

The Linux Foundation, 2019. *Kubernetes concepts - Namespace.* [Online]
Available at: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/
[Accessed 27 December 2019].

The Linux Foundation, 2019. *Kubernetes concepts - Nodes.* [Online]
Available at: https://kubernetes.io/docs/concepts/architecture/nodes/
[Accessed 27 December 2019].

The Linux Foundation, 2019. *Kubernetes concepts - Pods.* [Online]
Available at: https://kubernetes.io/docs/concepts/workloads/pods/pod-overview/
[Accessed 27 December 2019].

The Linux Foundation, 2019. *Kubernetes concepts - Service.* [Online]
Available at: https://kubernetes.io/docs/concepts/services-networking/service/
[Accessed 27 December 2019].

The Linux Foundation, 2019. *Kubernetes concepts - Volumes.* [Online]
Available at: https://kubernetes.io/docs/concepts/storage/volumes/
[Accessed 27 December 2019].

The Linux Foundation, 2019. *Kubernetes concpets - Controllers.* [Online]
Available at: https://kubernetes.io/docs/concepts/architecture/controller/
[Accessed 27 December 2019].