НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

«На правах рукопису»
УДК ____007.51_____

До захисту допущено:
В. о. завідувача кафедри
_____ Олександр РОЛІК
«___»_____2021 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інформаційне забезпечення робототехнічних систем»
зі спеціальності 126 «Інформаційні системи та технології»
на тему: «Автоматизована система управління бізнес-процесами страхової компанії»

Виконав:
студент VI курсу, групи ІК-01мп
Попенков Євген Андрійович _____

Керівник:
к.т.н., с.н.с.,
Тимошин Юрій Афанасійович _____

Рецензент:
Доцент каф. ІПІ, к.т.н.,доц.
Ліхоузова Тетяна Анатоліївна _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2021 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

ЗАТВЕРДЖУЮ
В. о. завідувача кафедри
_____ Олександр РОЛІК
«___»_____2021 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Попенкову Євгену Андрійовичу

1. Тема дисертації «Автоматизована система управління бізнес-процесами страхової компанії», науковий керівник дисертації Тимошин Юрій Афанасійович, к.т.н., с.н.с., затверджені наказом по університету від «27» 10 2021 р. № 3587-с

2. Термін подання студентом дисертації 10 грудня 2021р.

3. Об’єкт дослідження бізнес-процеси страхової компанії

4. Вихідні дані технології для автоматизації управління бізнес-процесами та технології для розробки веб-додатків

5. Перелік завдань, які потрібно розробити огляд існуючих рішень, проектування та розробка виконуючих схем бізнес процесів на базі веб-додатку, графічний матеріал

6. Орієнтовний перелік графічного (ілюстративного) матеріалу BPMN-схеми бізнес-процесів, ER-діаграма бази даних веб-додатку, архітектура роботи додатку.

7. Орієнтовний перелік публікацій

8. Консультанти розділів дисертації*

Розділ	Прізвище, ініціали та посада	Підпис, дата

* Якщо визначені консультанти. Консультантом не може бути зазначено наукового керівника магістерської дисертації.

	консультанта	завдання видав	завдання прийняв
Нормативний контроль	Пасько В.П., доцент		
Перевірка на співпадіння			

9. Дата видачі завдання 01.09.2021

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз бізнес-процесів страхової компанії	07.09.2021	
2	Огляд існуючих рішень.	14.09.2021	
3	Визначення проблем реалізації розроблюваної системи	21.09.2021	
4	Розробка бізнес моделі та опис основних бізнес-процесів	28.09.2021	
5	Розробка функціональних вимог до розроблюваної системи	05.10.2021	
6	Розробка архітектури системи (BPMN-схеми бізнес-процесів).	12.10.2021	
7	Опис реалізації елементів схеми бізнес-процесів	19.10.2021	
8	Підготовка та оформлення розділів роботи	25.10.2021	
9	Передзахист дисертації	25.11.2021	
10	Представлення до захисту	23.12.2021	

Студент Євген ПОПЕНКОВ

Науковий керівник Юрій ТИМОШИН

АНОТАЦІЯ

Метою даного проєкту є оптимізація бізнес процесів страхової компанії за допомогою їх автоматизації і представлення у вигляді веб-сервісу.

Система створена на базі веб-додатку, що забезпечить значну зручність та доступність. До неї було спроектовано базу даних веб-додатку.

В дипломному проєкті розроблено систему управління бізнес-процесами страхової компанії та веб-додатку, який надає доступ до даної системи. Також було розроблено базу даних, яка зберігає в собі користувачів, співробітників та їх взаємодії. Для реалізації системи були проаналізовані вже існуючі інструменти та технології та на їх базі було проведено аналіз та вибір основних технологій для розробки системи та веб-додатку. Також у рамках магістерської дисертації було описано та розроблено стартап-проєкт, тобто відповідну аргументовану документацію

Ключові слова: Java, BPMN, страхова компанія, оптимізація.

Розмір пояснювальної записки – 100 аркушів, містить 65 ілюстрацій, 23 таблиць, 8 додатків.

SUMMARY

The purpose of this project is to optimize the business processes of the insurance company through their automation and presentation in the form of a web service.

The system is based on a web application that will provide significant convenience and accessibility. A web application database was designed for it.

The diploma project developed a business process management system of the insurance company and a web application that provides access to this system. A database that also stores users, employees and their interactions has also been developed. To implement the system, the existing tools and technologies were analyzed and on their basis the analysis and selection of the main technologies for the development of the system and web application was carried out. Also in the framework of the master's dissertation was described and developed a startup project, ie the relevant reasoned documentation

Keywords: Java, BPMN, insurance company, optimization.The size of the explanatory note is 100 sheets, contains 65 illustrations, 23 tables, 8 appendices.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

BPMN(Business-Process Management Notation) – нотація управління бізнес процесів.

OSP (Open Source Project) – проект з відкритим кодом.

API (Application Programming Interface) – інтерфейс прикладного програмування.

HTTP (Hyper Text Transfer Protocol) – протокол передачі гіпертекстових документів.

WEB (World Wide Web) – міжнародна інформаційна система сховища інформації в електронному вигляді.

WEB-інтерфейс (Веб-інтерфейс) – сукупність засобів, за допомогою яких користувач взаємодіє з веб-сайтом або веб-застосунком через браузер.

REST (Representational State Transfer,) – підхід до архітектури мережевих протоколів, які забезпечують доступ до інформаційних ресурсів.

ВСТУП

На сьогоднішній день існує велика кількість підприємств, як малих так і великих, а в них в свою чергу відбувається багато різних бізнес-процесів, які частіше за все виконуються людиною або групою людей. Проте існують способи та інструменти для систематизації та автоматизації даних бізнес-процесів наприклад страхової компанії. Методологія BPMN допомагає автоматизувати та спростити як проєктування, так і підтримку та ведення робочих бізнес процесів.

Поняття BPM (Business Process Management) усе більш щільно входить в життя великих і малих корпорацій. Суть його в тому, щоб розглядати бізнес-процеси компанії, як активи, використовуючи які можна збільшити прибутковість св ого бізнесу. Інструмент, який ви для цього використовуєте, може бути будь-яким: аркуш паперу, текстовий документ, Visio або будь-який інший засіб створення діаграм. Але є клас інструментів, які спеціально призначені длятого, щоб послужити інструментом для трансформації вашого бізнесу, - це BPM-платформи. Завдання для такої платформи ставиться двояко: з одного боку необхідно візуалізувати бізнес-процес, а з іншого боку - його треба виконати.

Страховий бізнес – потужна сфера економіки. Професійним учасникам страхового ринку необхідно мати максимально наближене до реальності уявлення про стан справ у компанії та її майбутнє. Без цього неможливо планувати діяльність організації, здійснювати маркетингову стратегію, привертати інвестиції, не можливо розраховувати на успіх в повсякденних справах. Сьогодні керівництво страхових компаній починає проявляти все більший інтерес до нетрадиційних методологій і підходів, ситуаційного аналізу прийняття рішень, можливостей прогнозування нетривіальної поведінки бізнес-процесів на самих початкових фазах розвитку проектів. Більшість страховиків, як за кордом, так і в Україні, будують стратегію управління компанією, базуючись на процесному підході, оскільки, вважають, що саме цей підхід є необхідною умовою забезпечення стійкого і динамічного функціонування страхової організації. Страховиків цікавить розвиток всього бізнесу, тобто його функціонування на основі відповідних моделей бізнес-процесів, та пропозиції по їх вдосконаленню. Керівництво компаній бажає розуміння природи

ділових ситуацій, вивчення взаємодії проблемних ситуацій, ступеня їх дії на ефективність бізнес-процесів і плідність процедур прийняття стратегічних рішень. Для страховиків бізнес-процеси є найбільш пріоритетним завданням в порівнянні з іншими суб'єктами ринку фінансових послуг, оскільки характерною особливістю бізнес-процесів страхової компанії є наявність в них випадкових параметрів, обумовлених чинниками ризику. До їх числа можна віднести параметри, що характеризують страховий («виробничий») бізнес-процес (страхова премія, франшиза та ін.), адміністративний бізнес-процес (ефективність використання фінансових ресурсів, показники структури активів і пасивів та ін.), параметри, які знаходяться на перетині вказаних двох процесів (характеристики перестрахової та інвестиційної стратегій). Комплексний аналіз бізнес-процесів в страхуванні припускає також оцінку підсумкових показників діяльності страхової компанії (рентабельність, межа платоспроможності та ін.). Сучасний підхід до розробки бізнес-процесів страхової компанії має ідею постійного вдосконалення і модифікації, аналізу і прогнозування, а також своєчасного внесення змін в бізнес-моделі. Аналіз публікацій з проблем та сучасних технологій підтримки імітаційного моделювання дозволяє зробити висновок, що в даний час існує цілий ряд методик моделювання бізнес-процесів та платформ імітаційного моделювання для їх реалізації, які можливо застосовувати для отримання різного роду аналітичної інформації, отримання кількісних характеристик процесів, підвищення якості ухвалення управлінських рішень. Проте, цілий ряд питань, а саме, їх практичне застосування у бізнесзадачах, проведення аналізу поведінки бізнес-процесів, стратегічного планування найрізноманітніших управлінських ситуацій, дослідження особливостей моделей реінжинірингу та моделей забезпечення процесів реінжинірингу, потребує глибшого вивчення.

1 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Загальні терміни та положення

Управління бізнес-процесами (BPM) було визнано головним пріоритетом бізнесу та розбудовою бізнесу

Можливості процесу розглядаються як головна проблема для керівників вищого рівня в найближчі роки. BPM – це структурований, узгоджений і послідовний спосіб розуміння, документування, моделювання, аналіз, моделювання, виконання та безперервна зміна від кінця до кінця бізнес-процеси та всі залучені ресурси у світлі їхнього внеску в ефективність бізнесу.

Загалом, зростаючий попит на більш дисциплінований підхід до управління бізнес-процесами можна спостерігати, що спонукає багато організацій робити значні інвестиції в BPM ініціативи. Це, у свою чергу, викликало значну пов’язану академічну та комерційну роботу до передових рішень з управління бізнес-процесами. Одним із яскравих прикладів у цьому контексті є зростання популярності моделювання бізнес-процесів. З часом було запропоновано широкий спектр підходів до моделювання процесів, починаючи від простих блок-схеми до вдосконалених варіантів мереж Петрі з високою виразною силою. Один із найновіших/ Пропозиціями щодо ще однієї мови моделювання процесів є нотація моделювання бізнес-процесів (BPMN), версія 1.0 якого була запропонована в травні 2004 року та прийнята OMG для стандартизації цілей у лютому 2006 року. Відповідність новим стандартам веб-сервісів, його розумно інтуїтивне позначення та обіцянка стати офіційним галузевим стандартом моделювання процесів

підвищило популярність BPMN. Насправді, на офіційному веб-сайті BPMN вже є переліки понад 30 постачальників інструментів процесів, які підтримують BPMN. Увага, яку отримує BPMN, однак дотепер він не був збалансований критичним аналізом його реальних і уявних можливостей.

Дослідження, представлене в цій роботі, використовує принципи репрезентативного аналізу для дослідження потенційних і передбачуваних недоліків BPMN. Зокрема, він використовує модель представлення, запропоноване Вендом і Вебером (1995), яке спочатку було отримано з онтології визначена Bunge (1977), так звана модель

представлення BWW. За останнє десятиліття BWW модель була застосована до кількох провідних технік концептуального моделювання.

BPMI розробила або перебуває в процесі розробки три стандарти для полегшення BPM:

• BPMN, як стандарт моделювання бізнес-процесів,

• Мова моделювання бізнес-процесів (BPML), як стандартна мова виконання бізнес-процесів, та • Запит бізнес-процесів Мова (BPQL), стандартний інтерфейс керування для розгортання та виконання процесів електронного бізнесу. Знову ж таки, це аналогічно функціональності реляційних моделей даних і генерації операторів SQL/DDL. Мова моделювання бізнес-процесів (BPML) розроблена BPMI.org як стандартний опис бізнес-процесу на основі Pi-Calculus. BPMN дозволяє моделювати B2B і B2C. На відміну від попередніх типів діаграм бізнес-процесів, діаграма бізнес-процесів BPMN була створена з урахуванням мов бізнес-виконання та веб-сервісів. До діаграми були додані спеціальні позначення для зображення подій на основі повідомлень і

передача повідомлень між організаціями. Карти BPMN для мов ведення бізнесу. BPMN було визначено для безпосереднього відображення стандарту BPML та будь-яких інших конкуруючих мов виконання бізнесу, які впроваджуються, наприклад BPEL4WS, розробленого BEA, IBM, Microsoft та іншими.

Нотація моделювання бізнес-процесів (BPMN) — це мова візуального моделювання для додатків бізнес-аналізу та визначення робочих процесів підприємства, яка є відкритим стандартним позначенням для графічних блок-схем, що використовується для визначення робочих процесів бізнес-процесів. Це популярна та інтуїтивно зрозуміла графіка, яку можуть легко зрозуміти всі зацікавлені сторони бізнесу, включаючи бізнес-користувачів, бізнес-аналітиків, розробників програмного забезпечення та архітекторів даних.[1]

BPMN походить від синтезу кількох нотацій бізнес-моделювання. Спочатку опублікований Ініціативою управління бізнес-процесами (BPMI) у 2004 році, BPMN зараз підтримується OMG з моменту злиття двох організацій у 2005 році. BPMI об’єднався з OMG, Групою управління об’єктами. Документ специфікації BPMN

був опублікований OMG у лютому 2006 року. Версія 2.0 BPMN була розроблена в 2010 році, а фактична версія специфікації була випущена в грудні 2013 року. Останню версію (BPMN 2.0.2) офіційно опублікувала ISO як стандарт видання 2013 року: ISO/IEC 19510.

BPMN дозволяє фіксувати та документувати бізнес-процеси організацій чітким і послідовним способом, що забезпечує залучення до процесу відповідних зацікавлених сторін, таких як менеджери проектів та бізнес-користувачі. Таким чином, команда може ефективніше реагувати на будь-які проблеми, виявлені в процесі. BPMN надає вичерпні, але багаті позначення, які можуть легко зрозуміти як технічні, так і нетехнічні зацікавлені сторони. Моделювання бізнес-процесів надає важливі переваги таким компаніям і організаціям, як наведені нижче.

Галузевий стандарт, розроблений консорціумом OMG, некомерційною промисловою групою

Надає підприємствам можливість визначати та розуміти свої процедури за допомогою діаграм бізнес-процесів

Надати стандартне позначення, зрозуміле всім зацікавленим сторонам бізнесу

Щоб подолати комунікаційний розрив, який часто виникає між розробкою та впровадженням бізнес-процесів. Простий у навчанні, але досить потужний, щоб відобразити потенційні складності бізнес-процесу.

Огляд BPMN

Знання того, як працює бізнес, є першим і найважливішим кроком удосконалення бізнес-процесів. Модель і нотація бізнес-процесів (BPMN) забезпечує графічне представлення бізнес-процесів, яке може легко зрозуміти будь-хто, від бізнес-аналітика до зацікавленої сторони; допомога в аналізі бізнес-процесів і вдосконалення бізнес-процесів. Будь-який процес, описаний за допомогою BPMN, представляється як ряд кроків (діяльностей), які виконуються послідовно або одночасно відповідно до певних бізнес-правил. Подивіться на процес «Розмістити замовлення онлайн», який можна використовувати в інтернет-магазині, який розміщує замовлення в Інтернеті.[2]

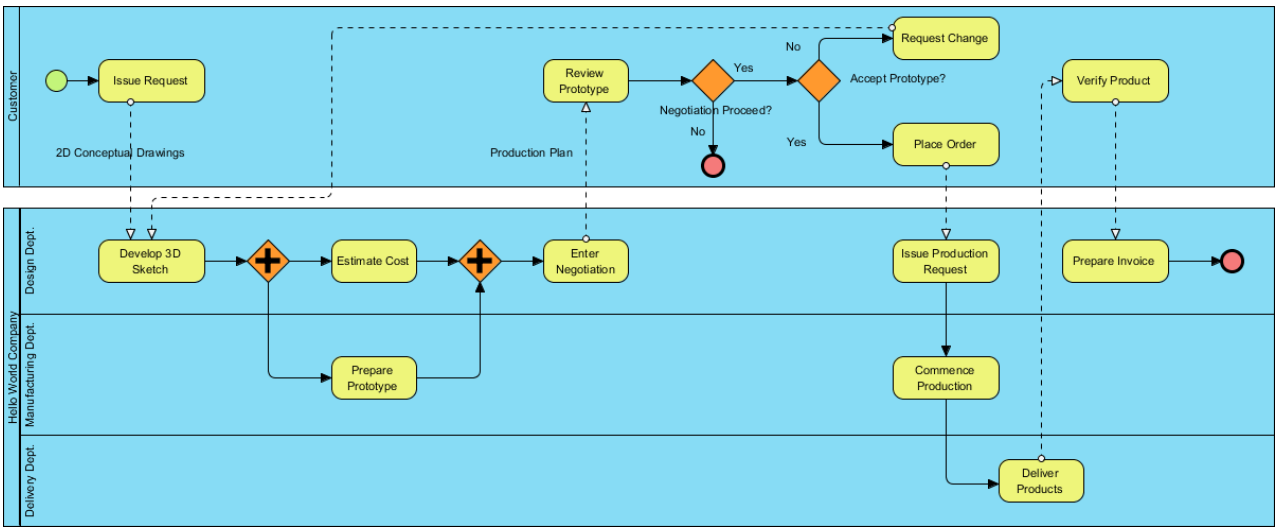

Рисунок 1.1 – Приклад діаграми бізнес-процесів

Нотація BPMN

Реалізації, які претендують на відповідність моделюванню процесів, повинні підтримувати такі пакети BPMN:

- Основні елементи BPMN;
- Діаграми процесів, які включають елементи, визначені в процесах, діяльності, даних і взаємодії людей
- Діаграми співпраці, які включають пули та потік повідомлень.
- Діаграми розмов, які включають пули, бесіди та посилання на бесіду.

В якості альтернативи повній відповідності моделювання процесів визначено три підкласи відповідності:

- Опис
- Аналітичний
- Загальний виконуваний файл

Описовий стосується видимих елементів і атрибутів, які використовуються в моделюванні високого рівня. Воно повинно бути зручним для

аналітики, які використовували інструменти блок-схеми BPA.

У BPMN процеси описуються за допомогою діаграм із серією графічних елементів. Така візуальна презентація дозволяє користувачам легко зрозуміти логіку процесу. BPMN в першу чергу був розроблений для проектування та читання як простих, так і складних діаграм бізнес-процесів. Для цього стандарт BPMN

класифікує графічні елементи за категоріями: в результаті елементи легко розпізнаються користувачами, які працюють з діаграмами бізнес-процесів.[2]

1.2 Огляд існуючих BPMN-інструментів

BPM – це неабиякою мірою підхід до ведення проекту, деякий образ мислення, а далеко не лише платформа для автоматизації написаного і погодженого ТЗ. І Pega, і IBM в цілому рекомендують схожі підходи: постановка цілей, ітеративна розробка, легкість змін; але різницю помітно в деталях. При усій схожості Pega і IBM, не можна проводити порівняння тільки базових платформ. У Pega є багато додаткових фреймворків, що ліцензуються окремо. У IBM є різні рівні ліцензування BPM, а також декілька інших платформ, доповнюючих BPM.

Огляд архітектури Pega:

Ядро Pega є java enterprise застосування, що запускається на будь-якому application сервері. На базі цього ядра побудований стек класів, складових базовий Framework PRPC, включаючи безпосередньо сам портал розробника, доступний через браузер (оскільки цей портал використовується і внутрішньою командою розробки, то можна сказати, що Pega розроблена на Pega).

Для розуміння того, як працює Pega, необхідно визначити два базові поняття: клас і правило :

Клас - це стандартна одиниця для об'єктно-орієнтованої парадигми, що є структурою, що містить деякі пов'язані дані і методи для їх обробки.

Правило - це екземпляр одного із спеціальних вбудованих (тобто визначених в одному з фреймворков) класів, що призначається або привласнюваний іншим класам. Правила можна розглядати як реалізацію патерну Стратегія (також іноді згадується як Behavior; повний опис патерну є на вики) : правило, що належить класу, визначає його властивість (property), поведінка (flow, flow action), спосіб відображення даних (section) і багато що інше.

Будь-який Pega-застосунок складається з набору класів, що визначають або структуру даних (тоді вони зберігаються в гілці Data-), або структуру робіт або кейсів (у гілці Work-). Кожен з цих класів може наслідувати властивості і методи

(визначувані правилами) від класів, визначених у фреймворках, або від інших класів застосування. Будь-який клас може перевизначити правило, задане у базовому класі.

Основа підходу базується на трьох речах:

Situational Layer Cake (Шарувата архітектура) : за рахунок спадкоємства і поліморфізму, реалізованих у вигляді Rule Resolution Mechanism, Pega дозволяє добитися того, щоб бізнес-процес для конкретного клієнта змінювався залежно від найрізноманітніших умов, наприклад, як від рівня лояльності цього клієнта, так і від регіонального законодавства обслуговуючого офісу.

6r - це концепція побудови комплексного рішення, яке забезпечує отримання (Receiving) і призначення (Routing) завдань, звітність (Reporting), реагування (Responding), збір інформації (Researching), ухвалення рішення і дозвіл (Resolving) кейсів.

DCO (Direct Capture of Objectives) - це методологія і набір підтримувальних її засобів, спрямовані на те, щоб у рамках проекту розроблялося тільки те, що реально треба бізнесу.

Огляд архітектури IBM:

Стеком платформ IBM є java enterprise застосування, що запускаються на сервері websphere application server.

IBM BPM має наступні елементи:

Визначення бізнес процесу - це виконувана діаграма процесу, відповідно до стандарту BPMN. Процес може включати інший процес або викликати конкретний сервіс.

Сервіс - атомарна виконувана одиниця застосування. Сервіси можуть бути вкладені в інші сервіси. Концептуально, сервіси бувають двох видів: повністю автоматизовані (виклик інтеграції, обробка даних і так далі) і неавтоматизовані (вимагаючі взаємодії з користувачем).

Тип даних - визначає структуру даних усередині застосування, може включати властивості простих типів (рядок, число і тому подібне) або складених, заданим

іншим типом даних. Конкретні змінні і екземпляри об'єктів даних визначаються в кожному процесі або сервісі.

IBM BPM складається з наступних частин:

Репозиторій процесів - зберігає усю інформацію про процеси, сервіси, структури даних і тому подібне;

Process server - виконуючий компонент, що забезпечує роботу бізнес-процесів;

Process center - компонент, що забезпечує доступ до репозиторія процесів для розробки;

Process designer - середовище розробки, грунтоване на eclipse;

Портал - середовище роботи кінцевих користувачів (може не використовуватися);

Process data warehouse - сховище статистичної інформації про проходження екземплярів процесів;

Blueworks Live - хмарний сервіс по моделюванню і документуванню бізнес-процесів. Побудована модель може бути імпортована в process designer, але на практиці ця можливість використовується рідко.

IBM BPM має наступні рівні ліцензій :

Express - містить усі описані вище компоненти BPM, але обмежений тільки одним проектом. Можливо встановити тільки на поодинокий сервер, без можливості кластеризації.

Standard - містить усі описані вище компоненти BPM. Дозволяє запускати відразу декілька проектів, дозволяє встановити на кластеризований сервер, базова підтримка інтеграції.

Advanced - містить додатково вбудовану шину даних і розширені засоби для інтеграції. Призначений для високо-навантажених проектів.

IBM має різні зв'язані продукти, для прикладу приведу деякі з них:

ODM - платформа для бізнес-правил;

Case manager - платформа для створення і організації набору кейсів;

Integration bus - інтеграційна шина.[3]

Camunda - це BPM-engine для автоматизації бізнес-процесів.

Camunda може дуже сильно, в десятки разів, понизити витрати на автоматизацію бізнес-процесів та/або оркестрацию мікросервісів. Це досягається за рахунок:
Camunda підтримує BPMN, нотацію опису бізнес-процесів. У BPMN можна намалювати логіку будь-якої складності, а движок її виконає. Більшість речей зв'язаних з процесами, які можна зробити в BPMN, робити розробляти самому дорожче в десятки разів.

Відкритий вихідний код дозволяє однозначно розуміти як працює програмне забезпечення, а відмінна документація дозволяє дуже швидко інтегрувати BPMN-engine у свою інфраструктуру.

Camunda підтримує останню версію Java, або будь-яку іншу мову програмування, яка базується JVM, наприклад Kotlin або Scala.
Також camunda являється зручним інструментом для розробки, тестування і вбудовування в CI\CD за рахунок того, що Camunda можна використовувати просто як бібліотеку в Java-застосунку. Також Camunda дозволяє інтегрувати в інші інструменти для різних цілей – статистичні аналізатори, тестові фреймворки, засобу контролю версій.

Наступна BPMN-технологія, яку слід розглянути – це фреймворк Activiti.
Основним компонентом фреймворка Activiti є механізм процесу. Механізм процесів надає основні можливості для виконання процесів Business Process Model and Notation (BPMN) 2.0 і створення нових завдань робочого процесу, серед іншого. Проект Activiti містить кілька інструментів на додаток до Activiti Engine.
і швидко пройдемося між різними компонентами. За допомогою Activiti Modeler аналітики бізнесу та інформації можуть моделювати бізнес-процеси, сумісні з BPMN 2.0, у веб-браузері. Це означає, що бізнес-процеси можна легко використовувати — не потрібно клієнтське програмне забезпечення, перш ніж ви зможете розпочати моделювання. Конструктор Activiti — це плагін на основі Eclipse, який дає змогу розробнику вдосконалити змодельований бізнес-процес у процес BPMN 2.0, який можна виконувати на механізмі процесів Activiti. Ви також можете запускати модульні тести, додавати логіку Java та створювати артефакти розгортання за допомогою Activiti Designer.

На додаток до інструментів проектування, Activiti надає ряд допоміжних інструментів. За допомогою Activiti Explorer ви можете отримати огляд розгорнутих процесів і навіть зануритися в таблиці бази даних під механізмом процесів Activiti. Ви також можете використовувати Activiti Explorer для взаємодії з розгорнутими бізнес-процесами. Наприклад, ви можете отримати список завдань, які вам уже призначені. Ви також можете запустити новий екземпляр процесу та переглянути стан цього новоствореного екземпляра процесу на графічній діаграмі.

Нарешті, є компонент Activiti REST, який надає веб-додаток, який запускає механізм процесу Activiti під час запуску веб-програми. Крім того, він пропонує REST API, який дозволяє віддалено спілкуватися з Activiti Engine.

Проект Activiti був започаткований у 2010 році Томом Баєєнсом і Йорамом Баррезом, колишнім засновником і основним розробником jBPM (JBoss BPM), відповідно. Метою проекту Activiti є створення надійного механізму процесу BPMN 2.0 з відкритим кодом. У наступному розділі ми детально розповімо про специфікацію BPMN 2.0, але в цьому розділі ми зосередимося на самій структурі Activiti та її інсталяції та запуску на простих прикладах.

Activiti фінансується Alfresco (відомою своєю однойменною системою управління документами з відкритим вихідним кодом; дивіться www.alfresco.com та главу 13, щоб дізнатися більше), але Activiti діє як незалежний проект з відкритим кодом. Alfresco використовує механізм процесів для підтримки таких функцій, як процес перегляду та затвердження документів, що означає, що документ має бути затверджений одним користувачем або групою користувачів. Для такого роду функціональних можливостей Activiti інтегровано в систему Alfresco, щоб забезпечити необхідні можливості механізму процесу та робочого процесу.

Примітка: jBPM використовувався в минулому замість Activiti, щоб забезпечити цей процес і функціональність робочого процесу. jBPM все ще включено в Alfresco, але в якийсь момент може бути застарілим.

Окрім запуску процесора Activiti в Alfresco, Activiti створений для автономної роботи або вбудованого в будь-яку іншу систему. У цій книзі ми зосередимося на

запуску Activiti за межами середовища Alfresco, але ми детально обговоримо можливості інтеграції між Activiti і Alfresco в розділі 13.

У 2010 році проект Activiti розпочався швидко і зумів випускати щомісячні (!) випуски фреймворка. У грудні 2010 року був доступний перший стабільний і готовий до виробництва випуск (5.0). Спільнота розробників Activiti, включаючи такі компанії, як SpringSource, FuseSource і Mulesoft, з тих пір часто розробляла нові функції. У цій книзі ми розглянемо цю внесену функціональність, таку як інтеграція Spring (розділ 4) та інтеграція Mule і Apache.

Activiti — це платформа процесів BPMN 2.0, яка реалізує специфікацію BPMN 2.0. Він здатний розгортати визначення процесів, запускати нові екземпляри процесів, виконувати завдання користувача та виконувати інші функції BPMN 2.0, які ми обговоримо в цій книзі.

Але за своєю суттю двигун Activiti є кінцевою машиною. Визначення процесу BPMN 2.0 складається з таких елементів, як події, завдання та шлюзи, які об’єднані разом за допомогою потоків послідовності. Коли таке визначення процесу розгортається на механізмі процесу і запускається новий екземпляр процесу, елементи BPMN 2.0 виконуються один за одним. Це виконання процесу подібне до кінцевого автомата, де є активний стан і, залежно від умов, виконання стану переходить в інший стан через переходи. У Activiti Engine більшість елементів BPMN 2.0 реалізовано як стан. Вони пов’язані з переходами, що виходять і прибувають, які в BPMN 2.0 називаються потоками послідовності. До кожного стану або відповідного елемента BPMN 2.0 може бути додана частина логіки, яка буде виконуватися, коли екземпляр процесу входить у стан. Як бачите, логічний інтерфейс ActivityBehavior реалізований багатьма класами. Це тому, що там реалізована логіка елемента BPMN 2.0.

Також слід розглянути таку систему BPMN як Flowable.

Flowable — це легкий бізнес-процес engine, написаний на Java. Механізм процесів Flowable дозволяє вам розгортати визначення процесів BPMN 2.0 (галузевий стандарт XML для визначення процесів), створюючи екземпляри процесів цих визначень, виконуючи запити, отримувати доступ до активних або

історичних екземплярів процесів і пов’язаних даних, а також багато іншого. У цьому розділі поступово будуть представлені різні концепції та API для цього за допомогою прикладів, які ви можете наслідувати на власній машині для розробки.

Flowable надзвичайно гнучкий, коли справа доходить до додавання його у вашу програму/послуги/архітектуру. Ви можете вбудувати двигун у свою програму або службу, включивши бібліотеку Flowable, яка доступна у вигляді JAR. Оскільки це JAR, ви можете легко додати його до будь-якого середовища Java: Java SE; контейнери сервлетів, такі як Tomcat або Jetty, Spring; Сервери Java EE, такі як JBoss або WebSphere тощо. Крім того, ви можете використовувати Flowable REST API для спілкування через HTTP. Існує також кілька Flowable додатків (Flowable Modeler, Flowable Admin, Flowable IDM і Flowable Task), які пропонують готові приклади інтерфейсу користувача для роботи з процесами та завданнями.

Загальним для всіх способів налаштування Flowable є основний механізм, який можна розглядати як набір служб, які надають API для керування та виконання бізнес-процесів.

Informatica ActiveVOS — це інструмент керування бізнес-процесами (BPM), який допомагає автоматизувати бізнес-процеси. Ви можете створювати моделі процесів, які інтегрують людей, процеси та системи, що підвищує ефективність вашого бізнесу.

Ви можете використовувати ActiveVOS, щоб переконатися, що оновлені дані об’єкта проходять робочий процес затвердження змін, перш ніж оновлені записи внесуть вклад у записи найкращої версії правди (BVT). Наприклад, бізнес-процес може вимагати, щоб старший менеджер переглядав та затверджував оновлення даних клієнтів, перш ніж вони стануть основними даними.

Щоб підтримувати робочий процес затвердження змін, MDM Hub і Data Director інтегруються з сервером ActiveVOS. Попередньо визначені робочі процеси, типи завдань і ролі MDM дозволяють компонентам синхронізуватися один з одним. Ви можете налаштувати свою реалізацію MDM для роботи з вбудованим сервером

ActiveVOS. Крім того, ви можете запустити окремий екземпляр ActiveVOS у своєму середовищі.

Вбудований ActiveVOS аутентифікує запити від Data Director і MDM Hub за допомогою певного принципала, якому довіряють як MDM, так і ActiveVOS. Цей принципал називається довіреним користувачем. Системний адміністратор створює облікові дані та ролі для довіреного користувача на сервері програм.

Сервер ActiveVOS має працювати на тому ж сервері додатків, що й MDM Hub. Додаткову інформацію див. у посібнику з конфігурації багатодоменного MDM.

Imixs-Workflow — це механізм робочого процесу з відкритим кодом, оптимізований для управління бізнес-процесами, орієнтованим на людину. Це переслідує такі цілі:

контролювати бізнес-процес і розподіляти завдання всередині організації на основі BPMN 2.0.

переконатися, що всі завдання обробляються відповідно до інструкцій та бізнес-правил.

надійно зберігати бізнес-дані та захищати їх від несанкціонованого доступу.

Таким чином, Imixs-Workflow покращує ваші бізнес-процеси та підтримує людські навички та діяльність, орієнтований на завдання та керований подіями.

Механізм Imixs-Workflow можна інтегрувати різними способами:

запустити диспетчер процесів Imixs за лічені секунди за допомогою Docker

після цього Imixs-Workflow інтегрується у програму Jakarta EE та розширюється її за допомогою архітектури мікроядра.

або ви можете запустити Imixs-Workflow як мікросервіс і взаємодіяти з двигуном через API Imixs-Rest.

Imixs-Workflow побудовано за стандартом Jakarta EE. У рамках цього стандарту бізнес-додатки можна розробляти на високомасштабованій та транзакційній структурі. Перегляньте посібник із швидкого запуску, щоб дізнатися, як працює механізм Imixs-Workflow у вашому власному бізнес-додатку. За допомогою API плагіна Imixs-Workflow ви можете розширити поведінку Imixs-Workflow.

Imixs-Workflow надає Rest Service API і може запускатися як сервіс в архітектурі мікросервісів. У цьому архітектурному стилі механізм робочого процесу може бути пов’язаний з будь-яким існуючим бізнес-додатком, незалежно від технологій, що стоять. Бізнес-логіку можна змінити, не змінюючи жодного рядка коду.

Ви можете почати з проекту Imixs-Microservice, який доступний на Github. Ви можете використовувати код як шаблон для власного мікросервісу. Проект Imixs-Workflow також підтримує використання Docker. За допомогою цієї технології ви можете запустити екземпляр Imixs-Workflow за допомогою однієї команди в контейнері. Дізнайтеся більше про те, як контейнеризувати Imixs-Workflow в розділі Docker. Дивіться також проект Imixs-Microservice на Github, який забезпечує повну підтримку Docker.

Далі буде розглянуто технологію jBPM.

BPM — це набір інструментів для створення бізнес-додатків для автоматизації бізнес-процесів і прийняття рішень.

jBPM походить від BPM (Управління бізнес-процесами), але він розвинувся, щоб дозволити користувачам вибирати власний шлях у автоматизації бізнесу. Він надає різноманітні можливості, які спрощують і перетворюють бізнес-логіку на багаторазові активи, такі як випадки, процеси, таблиці рішень тощо.

бізнес-процеси (BPMN2)

управління справами (BPMN2 і CMMN)

управління прийняттям рішень (DMN)

бізнес-правила (DRL)

оптимізація бізнесу (Solver)

jBPM можна використовувати як окрему службу або вбудувати в користувацьку службу. Він не вимагає використання жодної з фреймворків, його можна успішно використовувати в традиційних програмах JEE - розгортання війни/вуха

SpringBoot або Thorntail (раніше відомий як WildFly Swarm) - розгортання uberjar

автономні програми Java

jBPM зазвичай використовується для створення бізнес-додатків. Бізнес-додаток можна визначити як специфічне для домену рішення (створене з вибраними

фреймворками та можливостями), яке вирішує конкретну бізнес-задачу. Для реалізації бізнес-логіки він використовує можливості різних платформ, таких як бізнес-процеси, бізнес-правила та обмеження планування, а також стійкість, обмін повідомленнями, транзакції.

Бізнес-процес дозволяє моделювати ваші бізнес-цілі, описуючи кроки, які необхідно виконати для досягнення цієї мети та замовлення, використовуючи блок-схему. Це значно покращує наочність і гнучкість вашої бізнес-логіки, призводить до уявлень вищого рівня та для конкретних доменів, які можуть бути зрозумілі бізнес-користувачам і їх легше відстежувати.

Ядро jBPM — це легкий, розширюваний механізм робочих процесів, написаний на чистій Java, який дозволяє виконувати бізнес-процеси з використанням останньої специфікації BPMN 2.0. Він може працювати в будь-якому середовищі Java, вбудований у вашу програму або як сервіс.

На додаток до основного механізму пропонується безліч функцій та інструментів для підтримки бізнес-процесів протягом усього їхнього життєвого циклу:

Редактор на основі Eclipse і веб-редактор для підтримки графічного створення ваших бізнес-процесів і визначення регістрів (перетягування).

Підключаюча стійкість і транзакції на основі JPA / JTA.

Підключаюча служба завдань для людей на основі WS-HumanTask для включення завдань, які мають виконувати люди.

Консоль управління підтримує керування екземпляром процесів, списками завдань і формою завдань, а також звітуванням.

Додатковий репозиторій процесів для розгортання вашого процесу (та інших пов’язаних знань).

Реєстрація історії (для запитів / моніторингу / аналізу).

Інтеграція з різними фреймворками, такими як CDI/EJB, Spring(Boot), OSGi тощо.

BPM створює міст між бізнес-аналітиками, розробниками та кінцевими користувачами, пропонуючи функції та інструменти керування процесами, які подобаються як бізнес-користувачам, так і розробникам. Доменні вузли можуть бути

підключені до палітри, завдяки чому бізнес-користувачам буде легше зрозуміти процеси.

jBPM підтримує адаптивні та динамічні процеси, які вимагають гнучкості для моделювання складних реальних ситуацій, які неможливо легко описати за допомогою жорсткого процесу. Ми повертаємо контроль кінцевим користувачам, дозволяючи їм контролювати, які частини процесу мають виконуватися, динамічно відхилятися від процесу тощо.

jBPM також не просто ізольований механізм процесу. Складну бізнес-логіку можна змоделювати як поєднання бізнес-процесів з бізнес-правилами та складною обробкою подій. jBPM можна об’єднати з проектом Drools для підтримки єдиного середовища, яке інтегрує ці парадигми, де ви моделюєте свою бізнес-логіку як комбінацію процесів, правил і подій.

Висновок до розділу 1

В даному розділі розглянуто основні BPMN-технології на які слід звернути увагу при автоматизції бізнес-процесів.

Основними системами є Pega, IBM BPM та Camunda. Також було розглянуто основи архітектури вказаних вище технологій.

Недоліками вказаних рішень є: деякі затрати по часу для освоєння технології розробником, деякі з технологій не є проєктами з відкритим кодом, а також не мають безкоштовних ліцензій на використання продукту. Перевагами даних технологій є те, що вони дозволяють значно оптимізувати та полегшити проектування, розробку та подальше ведення автоматизованих бізнес-процесів. Слід відмітити, що продукти полегшують розробку та проектування не тільки розробникам, а й іншим спеціалістам таким, як бізнес-аналітики, менеджери проектів, а також дозволяє замовнику наглядно ознайомитись з розробленим функціоналом.[4]

2 ОПИС ОСНОВНИХ ПОНЯТЬ ТА ТЕХНОЛОГІЙ. РОЗРОБКА БІЗНЕС МОДЕЛІ

2.1 Опис та вибір BPMN-інструментів для розробки основних бізнес-процесів

Опис системи-прототипу:

В якості системи автоматизації бізнес процесів було вирішено створити web-додаток на базі bpmn-системи Camunda-BPMN та використати стек-технологій Spring.

BPMN визначає єдину діаграму бізнес-процесів, яка називається діаграмою бізнес-процесів (BPD). Ця діаграма була розроблена, щоб добре робити дві речі. По-перше, він простий у використанні та розумінні. Ви можете використовувати його для швидкого та легкого моделювання бізнес-процесів, і він легко зрозумілий нетехнічним користувачам (як правило, керівникам).
По-друге, він забезпечує виразність для моделювання дуже складних бізнес-процесів і може бути природним чином зіставлений з мовами виконання бізнесу. Щоб змоделювати потік бізнес-процесів, ви просто моделюєте події, які відбуваються для початку процесу, процеси, які виконуються, і кінцеві результати потоку процесу.

Бізнес-рішення та розгалуження потоків моделюється за допомогою шлюзів. Шлюз схожий на символ рішення на блок-схемі.
Крім того, процес у потоці може містити підпроцеси, які можуть бути графічно показані іншою діаграмою бізнес-процесів, підключеною через гіперпосилання до символу процесу. Якщо процес не розкладений на підпроцеси, він вважається завданням – процесом найнижчого рівня. Знак «+» у символі процесу означає, що процес розкладено; якщо на ньому немає позначки «+», це завдання.

Розмістивши події та процеси в спеціальних областях, які називаються пулами, які позначають, хто виконує процес.
Під час моделювання бізнес-процесів моделюються події, які відбуваються в бізнесі, і ілюструється, як вони впливають на потоки процесів. Подія або запускає потік процесу, або відбувається під час потоку процесу, або завершує потік процесу.

При моделюванні більш складних потоків процесів, такі як веб-сервіси B2B, потрібно моделювати більш складні бізнес-події, такі як повідомлення, таймери, бізнес-правила та умови помилок. BPMN дозволяє вказати тип тригера події та позначити його репрезентативним значком.

Визначення типу тригера для події накладає певні обмеження на потік процесу, який моделюється, що пояснюється в таблиці. Наприклад, таймер не може завершити потік процесу. Ви можете малювати лише потоки повідомлень від та до подій повідомлень.

Ці типи правил моделювання, які насправді є різновидами бізнес-правил, повинні застосовуватися автоматично інструментом моделювання, який підтримує BPMN. Часто подія відбувається під час виконання певного процесу, викликаючи переривання процесу і запускаючи новий процес. Або процес завершиться, що призведе до початку події та виконання нового процесу.

Можна змоделювати ці проміжні події, розмістивши символ події безпосередньо на процесі, з яким він пов’язаний.

В основі моделювання бізнес-процесів лежать самі процеси. Існує три типи процесів – процес, підпроцес і завдання. Кожен графічно зображений одним і тим же закругленим прямокутним символом; використання різних іменників просто відображає ієрархічні відносини між ними.

Процес намальований у вигляді закругленого прямокутника на діаграмі бізнес-процесів BPMN верхнього рівня. Ви можете вказати внутрішні деталі процесу, створивши або приєднавши до нього іншу діаграму бізнес-процесу. Піддіаграма вважається «дочірньою» діаграмою. Процес, який має дочірню діаграму, отримує "+"

маркер у його тілі. Графічне відображення деталей процесу за допомогою іншої діаграми бізнес-процесу вважається «декомпозицією» процесу. Процес можна розкласти без будь-яких обмежень - створення дочірньої діаграми для процесу і дочірніх діаграм для процесів на першій дочірній діаграмі тощо. Процеси, які зображені на «дочірніх» діаграмах, вважаються підпроцесами. Завданням вважається процес найнижчого рівня, який далі не розкладається.

Ця специфікація визначає ряд атрибутів і властивостей семантичних елементів, представлених графічними елементами, маркерами та зв'язками. Деякі з цих атрибутів є суто репрезентативними і так позначені, а деякі мають обов’язкові репрезентації. Деякі атрибути вказані як обов'язкові, але не мають представлення

або є лише необов'язковими. І деякі атрибути вказуються як необов’язкові. Для кожного атрибута чи властивості, які вказані як обов’язкові, відповідна реалізація повинна забезпечувати певний механізм, за допомогою якого значення цього атрибута чи властивості можуть бути створені та відображені. Цей механізм повинен дозволяти користувачеві створювати або переглядати ці значення для кожного елемента BPMN, який має цей атрибут або властивість. Якщо графічне представлення для цього атрибута або властивості вказано як вимагається, це графічне представлення має використовуватися. Якщо графічне представлення для цього атрибута або властивості вказано як необов’язкове, реалізація може використовувати або графічне представлення, або інший механізм. Якщо використовується графічне представлення, воно має бути зазначеним. Якщо для цього атрибута чи властивості не вказано графічне представлення, реалізація може використовувати або графічне представлення, або інший механізм. Якщо використовується графічне представлення, воно не має конфліктувати із зазначеним графічним представленням будь-якого іншого елемента BPMN.

Як згадувалося раніше, однією з цілей діаграми бізнес-процесів BPMN є можливість моделювання обміну повідомленнями B2B. З цією метою діаграма бізнес-процесів BPMN пропонує можливість моделювати потоки повідомлень. Традиційні діаграми бізнес-процесів дозволяють моделювати послідовні потоки процесів - від початкових подій до кінцевих результатів. Діаграма бізнес-процесів BPMN доповнює лінію Sequence Flow лінією Потік повідомлень, щоб ви могли моделювати людей або машини, які надсилають повідомлення один одному – важлива частина зображення та розуміння процесів між бізнесом і бізнесом та бізнес-споживачом.

BPMN визначає певні правила для моделювання потоків повідомлень і потоків послідовності. Потоки послідовності можуть бути намальовані лише між подіями, процесами та шлюзами в межах одного пулу. Потоки повідомлень можна оформляти лише між подіями, процесами або шлюзами, які існують у різних пулах, оскільки повідомлення передаються лише між різними організаціями чи програмами тощо. BPMN пропонує дотримуватись цих правил за допомогою інструмента

моделювання, який забезпечує підтримку BPMN. System Architect забезпечує виконання цих правил малювання, представляючи символ винищувача привидів і забороняючи з'єднання між неправильними елементами; він дозволяє лише підключати до відповідних елементів моделі. Це допомагає запобігти внесенню помилок або логічних невідповідностей у B2B-системи під час моделювання. Бувають випадки, коли ви моделюєте, що вам байдуже, як процес виконується в компанії. Це може бути інша компанія або клієнт. Компанію можна розглядати (або програму, функцію тощо) як «чорну скриньку» – проектуюстья лише потоки повідомлень до або з пулу, що її представляють, і не показуйте жодних деталі всередині басейну. Це на відміну від пулів, у яких ви моделюєте процеси, які можна вважати «білими ящиками» – ви можете зазирнути в них і вивчити їх деталі.

Інтернет – це неоднорідне середовище з безліччю різних платформ і додатків. У наскрізному ланцюжку створення вартості організації та окремі особи хочуть вибрати найкращі в своєму роді компоненти, які забезпечують ланцюжок вартості найкращу цінність. Програми та сервіси повинні працювати разом. Це одне з

рушійні сили для стандартизації веб-сервісів. Як ми згадували на початку цієї статті, робота веб-сервісів складається з чотирьох етапів: проектування процесів за допомогою BPMN, перевірка їх ефективності за допомогою моделювання, надання їх доступності шляхом публікації за допомогою мови виконання бізнес-процесів, а також оркестровка та координація. їх за допомогою системи управління бізнес-процесами (BPMS). BPMS пропонує можливість трансформувати окремі дисципліни робочого процесу, EAI та B2B із складного високоякісного рішення, яке практикують декілька висококваліфікованих спеціалістів.

консультантів у відкрите рішення, доступне для мас розробників, які створюють нові типи гнучких, слабо пов’язаних програм. BPMS організовує учасника (програми, людей, партнерів) у виконувані наскрізні процеси та закриває

розрив між стратегією та виконанням бізнесу. Ключові компанії, які розробляють BPMS, включають IBM, BEA Systems, Vitria, Intalio, FileNet, Fuego і Collaxa. Важливість оцінки доступних методів для моделювання зростає зі зростанням кількості доступних методів, оскільки результати допоможуть

користувачам вибрати найбільш відповідний метод для поставленої задачі. Традиційно дослідницька спільнота зосереджується на створенні нових мов моделювання, а не на оцінці тих, які є

вже існує.

Оцінюючи існуючі методи, можна не тільки порівняти їх придатність для вирішення поставленої задачі, але це також допоможе визначити навички, необхідні від користувача та модельної аудиторії, перш ніж взятися за завдання моделювання.

Використовуючи формалізовані рамки при оцінці нових методів і порівнюючи дані з результатами попередніх досліджень, можна було б визначити, чи є загальний рейтинг нового методу вищим, ніж його попередники.

Різні підходи до оцінки мов моделювання включають аналітичні та емпіричні методи, існують як одномовні, так і порівняльні оцінки. Емпіричні методи повинні досліджувати як можливість моделювання використовувати мову, розуміння моделей, розроблених мовою, так і здатність вчитися та діяти відповідно до знань, наданих у моделях. Хоча аналітичні оцінки можуть бути проведені, щойно специфікація мови стане доступною, емпіричні оцінки в більшості випадків вимагають від користувачів нового методу певного досвіду його використання, а для цього методу знадобиться деякий час з спільноти користувачів до того, як оцінка може відбутися. Емпіричні дослідження можуть включати дослідження того, чи підтверджуються результати аналітичних досліджень і в якій мірі

вони впливають на практиці. Це також включатиме проведення тематичних досліджень та опитувань, щоб виявити, чи є метод настільки прийнятним, як очікувалося, і чи використовується він відповідно до очікувань.

BPMN більше не вважається новим, і його оцінили як аналітично, так і емпірично.

Workflow Patterns Framework2 надає таксономію загальних, повторюваних концепцій і конструкцій, релевантних у контексті інформаційних систем, що обізнані з процесом. Шаблони робочого процесу описують ядро базових структур, які можна було б очікувати від підтримки систем робочого процесу. Визначення цих

закономірностей дало змогу порівняти виразну силу наявних комерційних інструментів для бізнес-процесу

моделювання. Пізніше шаблони були визнані застосовними в набагато ширшому сенсі, і вони були використані для вивчення можливостей мов моделювання бізнес-процесів, таких як BPMN, діаграми активності UML та EPC; мови компонування веб-сервісів, такі як WCSI; і мови виконання бізнес-процесів, такі як

як BPML, XPDL і BPEL. Доступні шаблони поділяються на перспективу потоку керування, перспективу даних та перспективу ресурсу. Оригінальні візерунки складалися з 20, 40 і 43 шаблонів відповідно. Перегляд моделей потоку контролю, проведений у 2006 році, призвів до додаткових 23 моделей. Наведено три причини вибору шаблонів робочого процесу

Recker. Це добре прийнята структура, яка широко використовується

як для вибору систем управління робочим процесом, так і для самооцінки постачальником продуктів моделювання процесів; Фреймворк зарекомендував себе в індустрії, запустив розширення для систем моделювання процесів і надихнув їх розвиток. Результати оцінок включають:

Представництво держави. Через відсутність представлення стану в BPMN виникають труднощі з представленням певних моделей потоків керування. Існують додаткові притаманні труднощі із застосуванням шаблонів робочих процесів для оцінки мови, яка не має загальноузгодженої формальної семантики чи середовища виконання. Відображення BPEL, надане в BPMN

специфікація є лише частковою, залишаючи осторонь моделі з неструктурованими топологіями. У специфікації BPMN можна знайти кілька неясностей через відсутність формалізації (.

Кілька зображень одного і того ж шаблону. Прості шаблони робочого процесу мають кілька представлень BPMN, а для захоплення найдосконаліших шаблонів потрібно глибоке знання атрибутів, пов’язаних з моделюючими конструкціями BPMN, які не мають графічного представлення. Підтримка екземплярів. Шаблони

даних робочого процесу та середовища не підтримуються через відсутність підтримки даних, що стосуються екземпляра, для завдання або підпроцесу

з маркером «кілька екземплярів» не можна вказати.

Ресурсне моделювання. Підтримка ресурсної перспективи в BPMN мінімальна, але моделювання організаційних структур і ресурсів розглядається як поза межами BPMN. Автори стверджують, що конструкції доріжок і басейнів суперечать цьому. Порівняйте виразну силу доступних комерційних інструментів для бізнес-процесів

моделювання. Пізніше шаблони були визнані застосовними в набагато ширшому сенсі, і вони були використані для вивчення можливостей мов моделювання бізнес-процесів, таких як BPMN, діаграми активності UML та EPC; мови компонування веб-сервісів, такі як WCSI; і мови виконання бізнес-процесів, такі як BPML, XPDL і BPEL.

Доступні шаблони поділяються на перспективу потоку керування, перспективу даних та перспективу ресурсу. Оригінальні візерунки складалися з 20, 40 і 43 шаблонів відповідно. Перегляд моделей потоку контролю, проведений у 2006 році, призвів до додаткових 23 моделей. Це добре прийнята структура, яка широко використовується як для вибору систем управління робочим процесом, так і для самооцінки постачальником продуктів моделювання процесів; Фреймворк зарекомендував себе в індустрії, і він викликав розширення для систем моделювання процесів і надихнувїх розвиток. Через відсутність представлення стану в BPMN виникають труднощі з представленням певних моделей потоків керування. Існують додаткові притаманні труднощі із застосуванням шаблонів робочих процесів для оцінки мови, яка не має загальноузгодженої формальної семантики чи середовища виконання.

Як основне BPMN-рішення було вирішено обрати Camunda, так як дана технологія є рішенням, яке має достатньо гнучкий безкоштовний функціонал для автоматизації бізнес-процесів. Також Camunda надає різні можливості для зручного розгортання веб-застосунків та інтеграції корисних технологій, як для розробки так і для подальшого підтримання проєкту.

Також для зручності та швидкості розробки та розгортання веб-додатку було обрано стек технологій або фреймворк Spring.

Компоненти стеку, який надається Camunda:

Camunda — це набір застосунків: Modeler, Task List, BPMN Engine, DMN Engine, Cockpit, Admin,Optimize.

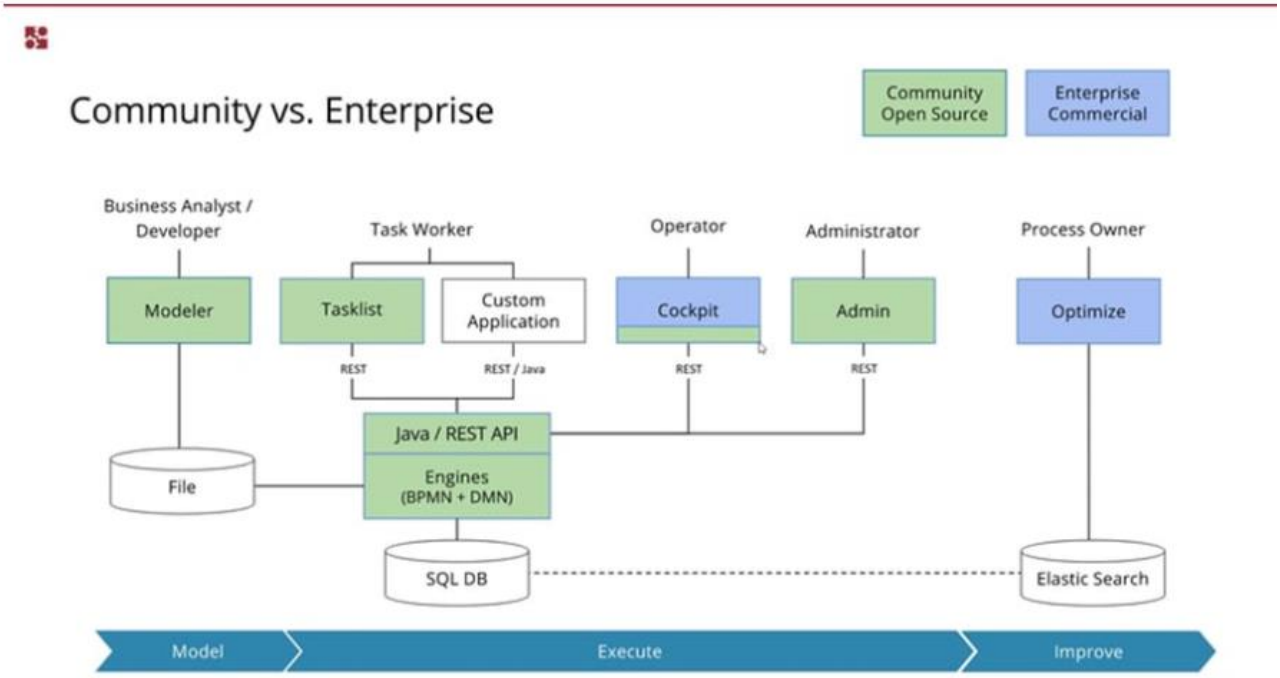

Рисунок 2.1 – Архитектура застосунків Camunda

Modeler— це застосунок для створення моделей BPMN процесів. Ці моделі потрібні для інших частин системи.

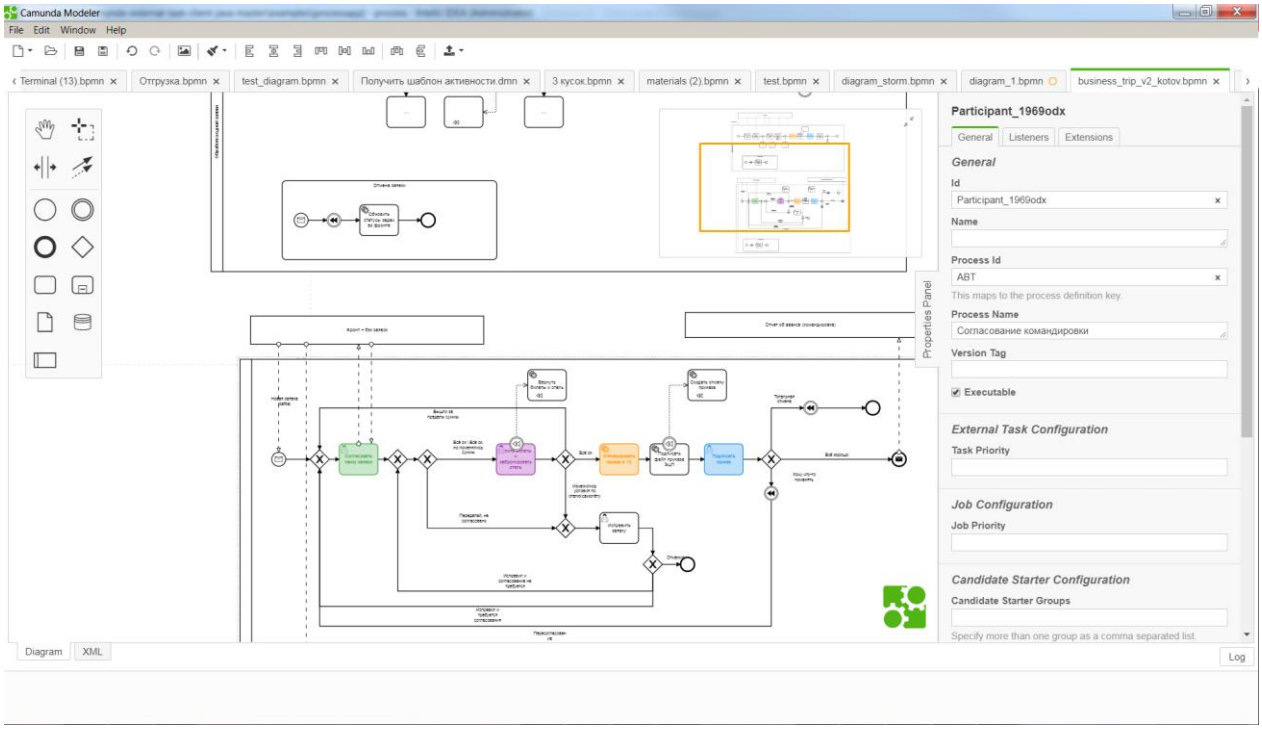

Рисунок 2.2 – Інтерфейс Camunda Modeler

Task list — це веб-застосунок, в якому кінцевий користувач (робітники страхової компанії) виконують задачі, Які відводяться їм бізнес-процесом.

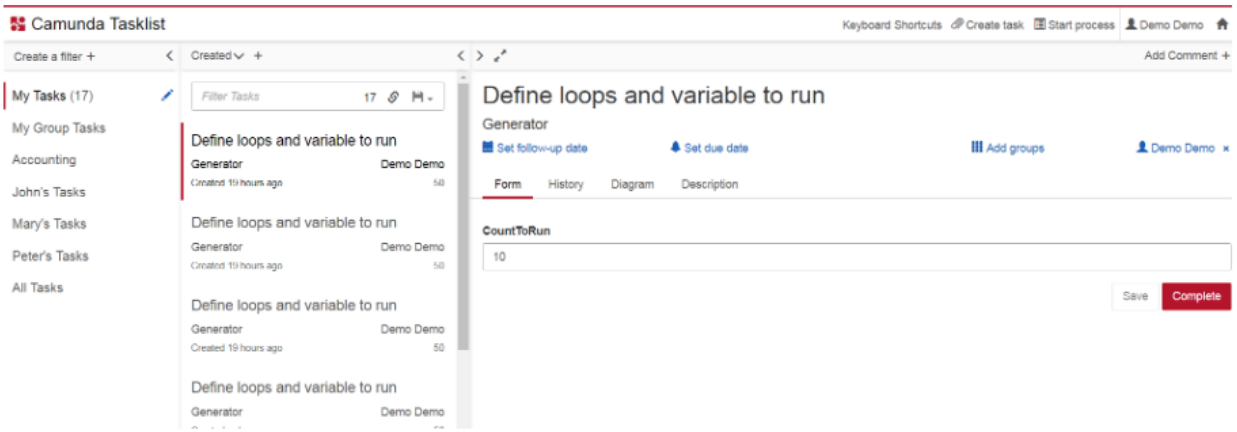

Рисунок 2.3 – Інтерфейс camunda tasklist

BPMN Engine — це безпосередньо «двигун» (engine), який відповідає за інтерпретацію BPMN в объекти мови програмування Java, збереження об’єктів в базі даних та реалізацію інших технічних завдань.

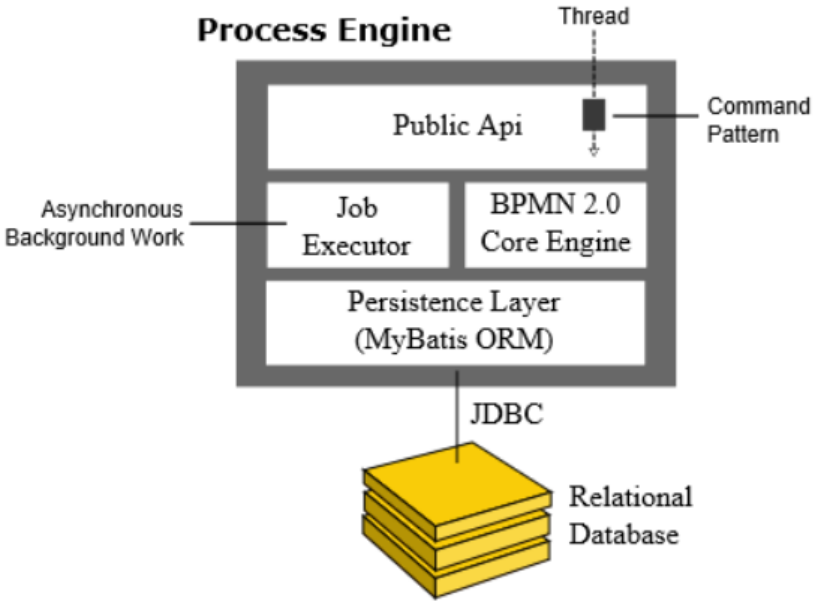

Рисунок 2.4 Структура BPMN-engine

DMN Engine — аналогічно BPMN Engine, тільки для DMN-схем.

Cockpit — це веб-застосунок для перегляду стану процесів.

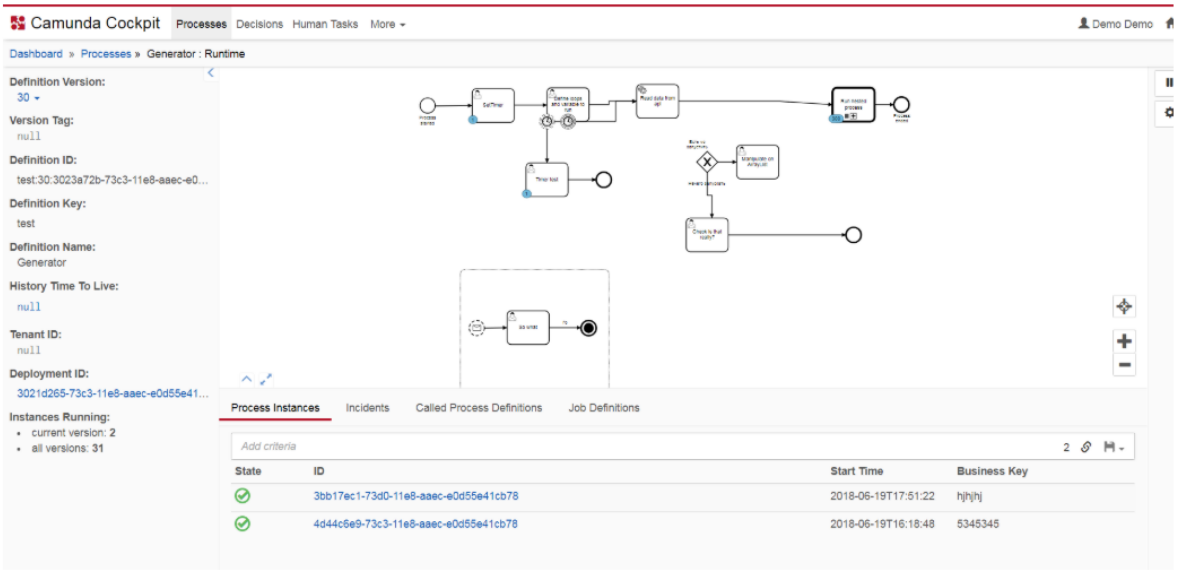

Рисунок 2.5 – Інтерфейс Camunda Cockpit

Admin — це веб-застосунок для управління правами користувачів та самими користувачами відповідно.

Optimize — це веб-застосунок для аналізу бізнес-процесів.

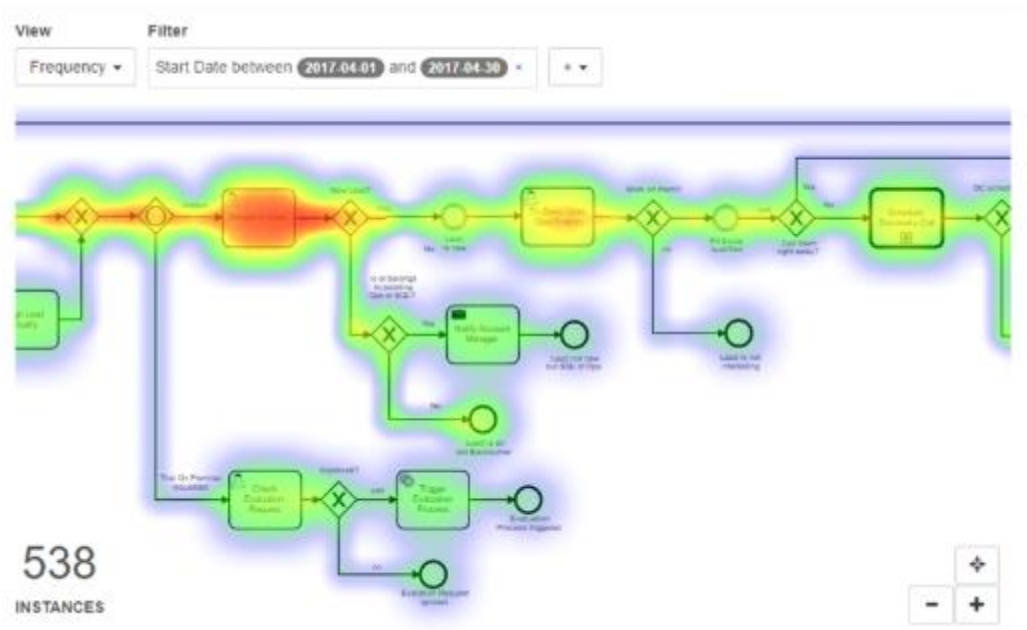

Рисунок 2.6 – Інтерфейс Camunda Optimize (демонстрація навантаженості процесу).

Для опису та розробки достатньо буде застосувати Camunda Modeler, Cockpit, TaskList, так як Optimize входить лише у платний пакет Camunda Enterprise, а також потрібен для аналізу та статистики, тому він вже може знадобитись на стадії ведення та підтримки проєкту.[5]

2.2 Основні поняття Camunda-BPM

Завдання (Task) – дозволяють моделювати фактичну роботу, що виконується в процесі. Підтримуються різні типи завдань.

Сервісне завдання(Service Task) використовується для виклику служб. У Camunda це робиться шляхом виклику коду Java або надання робочого елемента для зовнішнього працівника для асинхронного завершення або виклику логіки, яка реалізується у формі веб-сервісів.

Рисунок 2.7 – Позначення сервісного завдання на BPMN-схемі

Завдання користувача (User Task) використовується для моделювання роботи, яку має виконати людина. Коли виконання процесу досягає такої Завдання користувача, у списку завдань користувача (-ів) або групи (груп), призначених для цього завдання, створюється нове завдання.

Рисунок 2.8 – Позначення завдання користувача на BPMN-схемі

Для відправки повідомлення використовується завдання надсилання (Send Task). У Camunda це робиться за допомогою виклику коду Java.

Завдання надсилання має таку ж поведінку, як і службове завдання.

Рисунок 2.9 – Позначення завдання надсилання на BPMN-схемі

Шлюз з виключенням (також званий шлюзом XOR або, більш технічно, ексклюзивним шлюзом на основі даних) використовується для моделювання рішення в процесі. Коли виконання надходить на цей шлюз, усі вихідні потоки послідовностей оцінюються в тому порядку, в якому вони були визначені. Для продовження процесу вибирається потік послідовності, умова якого оцінюється як «істина» (або який не має набору умов, який концептуально має значення «істина», визначене в потоці послідовності). Слід зауважити, що при використанні ексклюзивного шлюзу вибирається лише один потік послідовності. У випадку, якщо потік кількох послідовностей має умову, яка оцінюється як «істинна», для продовження процесу вибирається виключно перший. Якщо жоден потік послідовності не можна вибрати (жодна умова не має значення «істина»), це призведе до винятку під час виконання, якщо не визначено потік за замовчуванням. Один потік за замовчуванням може бути встановлений на самому шлюзі, якщо жодна інша умова не відповідає — як «інший» у мовах програмування.[6]

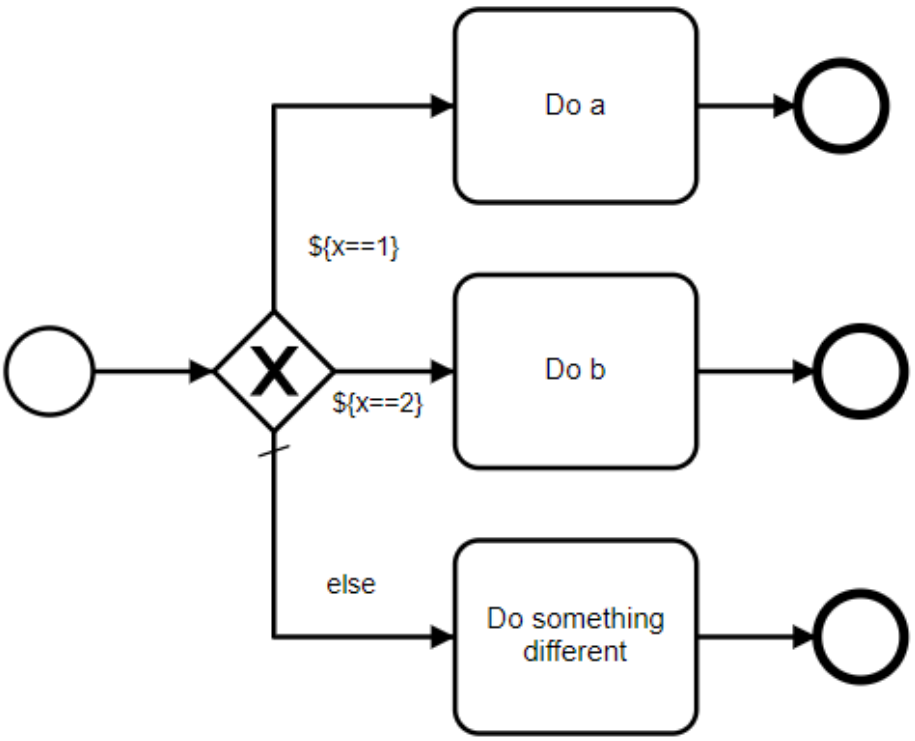

Рисунок 2.10 – Приклад BPMN-схеми з використанням шлюзу з виключенням на основі даних

Шлюзи також можна використовувати для моделювання паралельності в процесі. Найпростішим шлюзом для введення паралельності в модель процесу є паралельний шлюз, який дозволяє розгалужуватися на кілька шляхів виконання або об’єднувати кілька вхідних шляхів виконання.

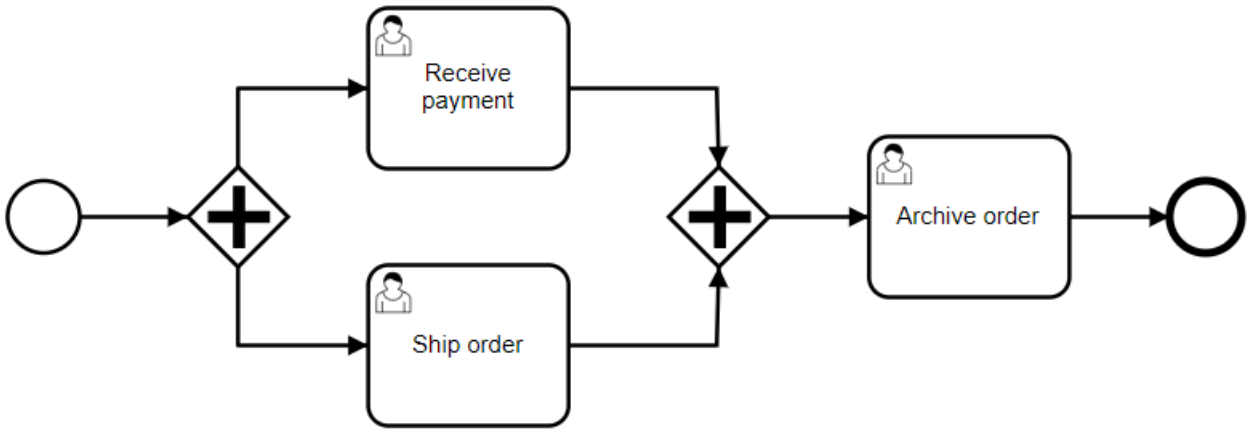

Рисунок 2.11 – Приклад BPMN-схеми з використанням паралельного шлюзу

Функціональність паралельного шлюзу заснована на вхідних і вихідних потоках послідовностей: всі вихідні потоки послідовності виконуються паралельно, створюючи одне одночасне виконання для кожного потоку послідовності.

приєднатися: усі одночасні виконання, що надходять на паралельний шлюз, чекають на шлюзі, доки не надійде виконання для кожного з потоків вхідної послідовності. Потім процес продовжується після приєднання шлюзу.

Слід зауважтити, що в реалізації паралельного шлюзу Camunda шлюз запускається, як тільки виконується наступне: кількість маркерів, що надійшли, дорівнює кількості вхідних потоків послідовності. Не обов’язково, щоб маркер надходив на кожен вхідний потік. Також слід відмтити, що паралельний шлюз може мати поведінку як розгалуження, так і з’єднання, якщо для одного паралельного шлюзу є кілька вхідних і вихідних потоків послідовності. У цьому випадку шлюз спочатку об’єднає всі вхідні потоки послідовності, а потім розділить на кілька одночасних шляхів виконання. Важливою відмінністю від інших типів шлюзів є те, що паралельний шлюз не оцінює умови. Якщо умови визначені для потоку послідовності, пов’язаного з паралельним шлюзом, вони просто ігноруються.

Шлюз **з** включенням (інклюзивний) можна розглядати як комбінацію шлюзу з виключенням та паралельного шлюзу. Як і у випадку із шлюзом з виключенням, можна визначити умови для вихідних потоків послідовності. Однак основна відмінність полягає в тому, що шлюз з включенням може приймати більше одного потоку послідовності, як паралельний шлюз. Функціональність інклюзивного шлюзу заснована на вхідних і вихідних потоках послідовності:

fork: оцінюються всі умови потоку вихідної послідовності, а для умов потоку послідовності, які оцінюються як «істина», потоки слідують паралельно, створюючи одне одночасне виконання для кожного потоку послідовності.

join: усі одночасні виконання, що надходять на інклюзивний шлюз, чекають на шлюзі, доки не надійде виконання для кожного з потоків вхідної послідовності, які мають маркер процесу. Це важлива відмінність від паралельного шлюзу. Іншими словами, інклюзивний шлюз буде чекати лише вхідних потоків послідовності, які виконуються. Після приєднання процес продовжується після приєднання інклюзивного шлюзу.

Слід зауважити, що інклюзивний шлюз може мати поведінку як розгалуження, так і приєднання, якщо для одного інклюзивного шлюзу існує кілька вхідних і вихідних потоків послідовності. У цьому випадку шлюз спочатку об’єднає всі вхідні потоки послідовності, які мають маркер процесу, а потім розділить на кілька одночасних шляхів виконання для вихідних потоків послідовності, які мають умову, яка оцінюється як «істина».

Стандартна подія початку або «пуста стартова подія» є невизначеними подіями, які також називаються «порожніми» подіями. Стандартна подія початку технічно означає, що тригер для запуску екземпляра процесу не визначений. Це означає, що двигун не може передбачити, коли потрібно запустити екземпляр процесу. Подія none start використовується, коли екземпляр процесу запускається через API

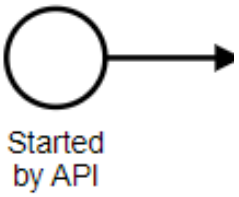

Рисунок 2.12 – Позначення стандартного старту процесуу на BPMN-схемі

Подія початку з повідомленням може використовуватися для запуску екземпляра процесу за допомогою іменованого повідомлення. Це ефективно дозволяє вибрати правильну подію початку з набору альтернативних подій початку, використовуючи назву повідомлення. Під час розгортання визначення процесу з одним або кількома подіями початку повідомлення застосовуються такі міркування:

Ім’я події початку повідомлення має бути унікальним для даного визначення процесу, тобто визначення процесу не повинно мати кількох подій початку повідомлення з однаковою назвою. Механізм створює виняток під час розгортання визначення процесу, якщо дві або більше подій початку повідомлення посилаються на одне повідомлення або якщо два або більше подій початку повідомлення посилаються на повідомлення з однаковою назвою повідомлення.

Ім’я події початку повідомлення має бути унікальним для всіх визначень розгорнутих процесів. Механізм створює виняток під час розгортання визначення процесу, якщо одна або кілька подій початку повідомлення посилаються на повідомлення з такою ж назвою, що й подія початку повідомлення, уже розгорнута іншим визначенням процесу.

Управління версіями процесу: після розгортання нової версії визначення процесу підписка на повідомлення попередньої версії скасовується. Це також стосується подій повідомлень, яких немає в новій версії.

Процес можна запустити за допомогою одного з двох різних повідомлень. Це корисно, якщо процесу потрібні альтернативні способи реагування на різні події початку, але в кінцевому підсумку він продовжується однаково.

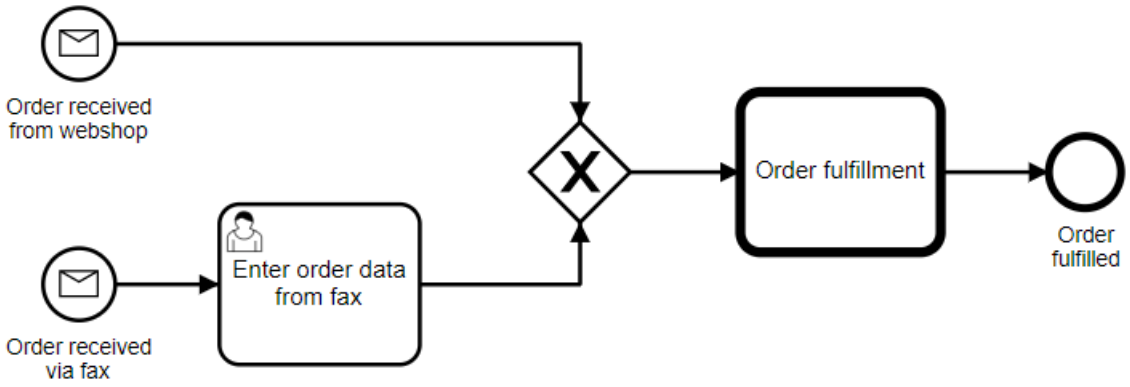

Рисунок 2.13 – Приклад процесу з двома подіями початку з повідомлення на BPMN-схемі

Підпроцес — це діяльність, яка містить інші дії, шлюзи, події тощо, яка сама по собі утворює процес, що є частиною більшого процесу. Підпроцес повністю визначається всередині батьківського процесу (тому його часто називають вбудованим підпроцесом).

Підпроцеси мають два основних варіанти використання:

Підпроцеси дозволяють ієрархічне моделювання. Багато інструментів моделювання дозволяють згортати підпроцеси, приховуючи всі деталі підпроцесу та відображаючи наскрізний огляд бізнес-процесу високого рівня. Підпроцес створює нову область для подій. Події, які генеруються під час виконання підпроцесу, можуть бути перехоплені граничною подією на кордоні підпроцесу, створюючи таким чином область для цієї події, обмежену підпроцесом.

Використання підпроцесу накладає деякі обмеження:

Підпроцес може мати лише одну стандартну подію початку, інші типи подій початку не дозволені. Підпроцес повинен мати принаймні одну кінцеву подію. Зауважте, що специфікація BPMN 2.0 дозволяє опускати події початку та кінця підпроцесу, але поточна реалізація двигуна цього не підтримує.

Потоки послідовності не можуть перетинати межі підпроцесу. Підпроцес візуалізується як типова діяльність, тобто закруглений прямокутник. У разі згортання підпроцесу відображаються лише ім’я та знак плюса, що дає повний огляд процесу:

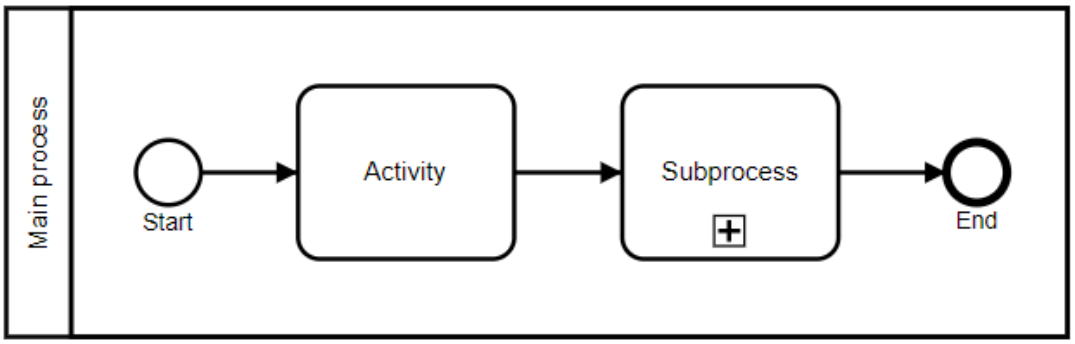

Рисунок 2.14 – Приклад BPMN-схеми зі згорнутим підпроцесом

Якщо підпроцес розгорнутий, кроки підпроцесу відображаються в межах підпроцесу:

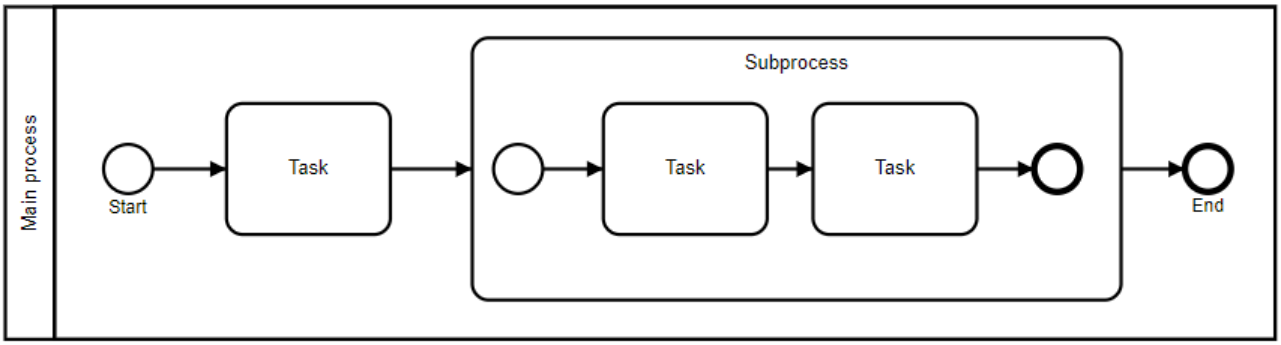

Рисунок 2.15 – Приклад BPMN-схеми з розгорнутим підпроцесом

Однією з основних причин використання підпроцесу є визначення області дії для події. І залежно від різних подій можна «спрямовувати» процес тим чи іншим шляхом:

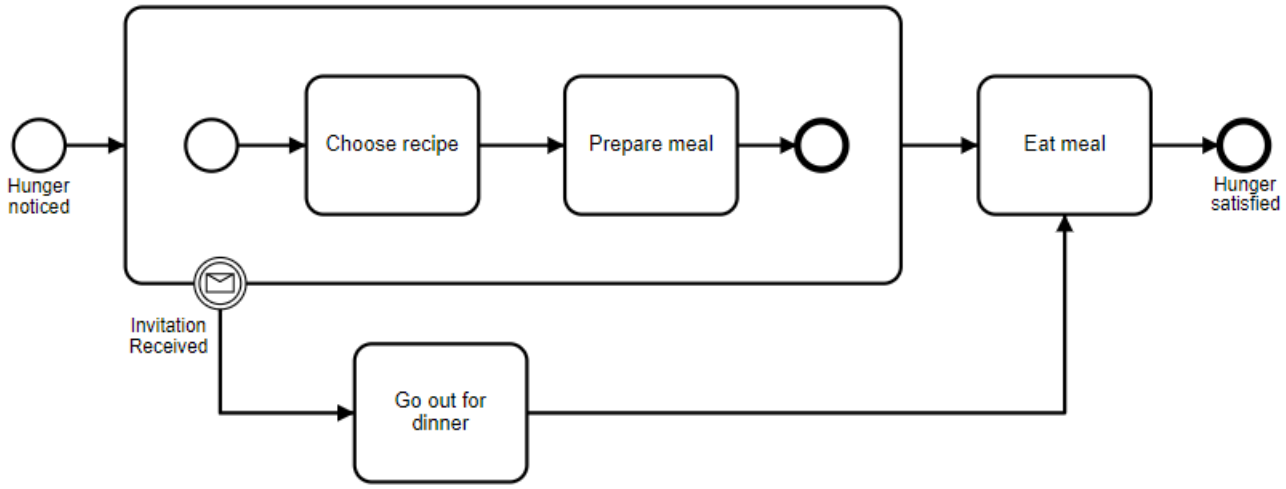

Рисунок 2.16 – Приклад BPMN-схеми з підпроцесом та альтернативним шляхом у вигляді події з прийомом повідомлення

BPMN 2.0 розрізняє вбудований підпроцес і активність виклику. З концептуальної точки зору, обидва викличуть підпроцес, коли виконання процесу приходить до дії. Різниця полягає в тому, що активність виклику посилається на процес, який є зовнішнім щодо визначення процесу, тоді як підпроцес вбудований у вихідне визначення процесу. Основний випадок використання для дії виклику — мати багаторазове визначення процесу, яке можна викликати з кількох інших визначень процесу.

Коли виконання процесу приходить до дії виклику, створюється новий екземпляр процесу, який використовується для виконання підпроцесу, потенційно створюючи паралельні дочірні виконання, як у звичайному процесі. Екземпляр основного процесу чекає, поки підпроцес повністю завершиться, а потім продовжує початковий процес.

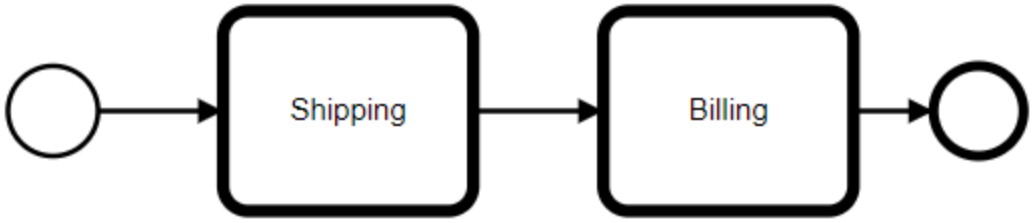

Рисунок 2.17 – Приклад BPMN-схеми з підпроцесами з викликом інших активностей (інші процеси або зовнішні сервіси)

Активність виклику візуалізується так само, як і згорнутий вбудований підпроцес, але з товстою рамкою. Залежно від інструмента моделювання, активність виклику також може бути розгорнута, але візуалізація за замовчуванням — це згорнуте уявлення.

Підпроцес транзакції — це вбудований підпроцес, який можна використовувати для групування кількох дій у транзакцію. Транзакція — це логічна одиниця роботи, яка дозволяє групувати набір окремих видів діяльності, щоб вони або були успішними, або невдалими разом. Транзакція може мати три можливі результати:

Транзакція вважається успішною, якщо її не скасовують і не припиняють небезпеки. Якщо підпроцес транзакції пройшов успішно, він залишається з використанням вихідного потоку послідовності. Успішна транзакція може бути компенсована, якщо компенсаційна подія буде викликана пізніше в процесі.

Примітка: як і «звичайні» вбудовані підпроцеси, підпроцес транзакції може бути компенсований після успішного завершення за допомогою проміжної події компенсації викиду. Транзакція скасовується, якщо виконання досягає події завершення скасування. У цьому випадку всі виконання припиняються та видаляються. Після цього для одного виконання, що залишилося, встановлюється гранична подія скасування, яка запускає компенсацію. Після завершення компенсації підпроцес транзакції залишається, використовуючи вихідний потік послідовності граничної події скасування. Транзакція завершується небезпекою, якщо виникає подія помилки, яка не вловлюється в межах підпроцесу транзакції. (Це також застосовується, якщо помилка виявляється на межі підпроцесу транзакції.) У цьому випадку компенсація не виконується. Наступна діаграма ілюструє три різні результати:

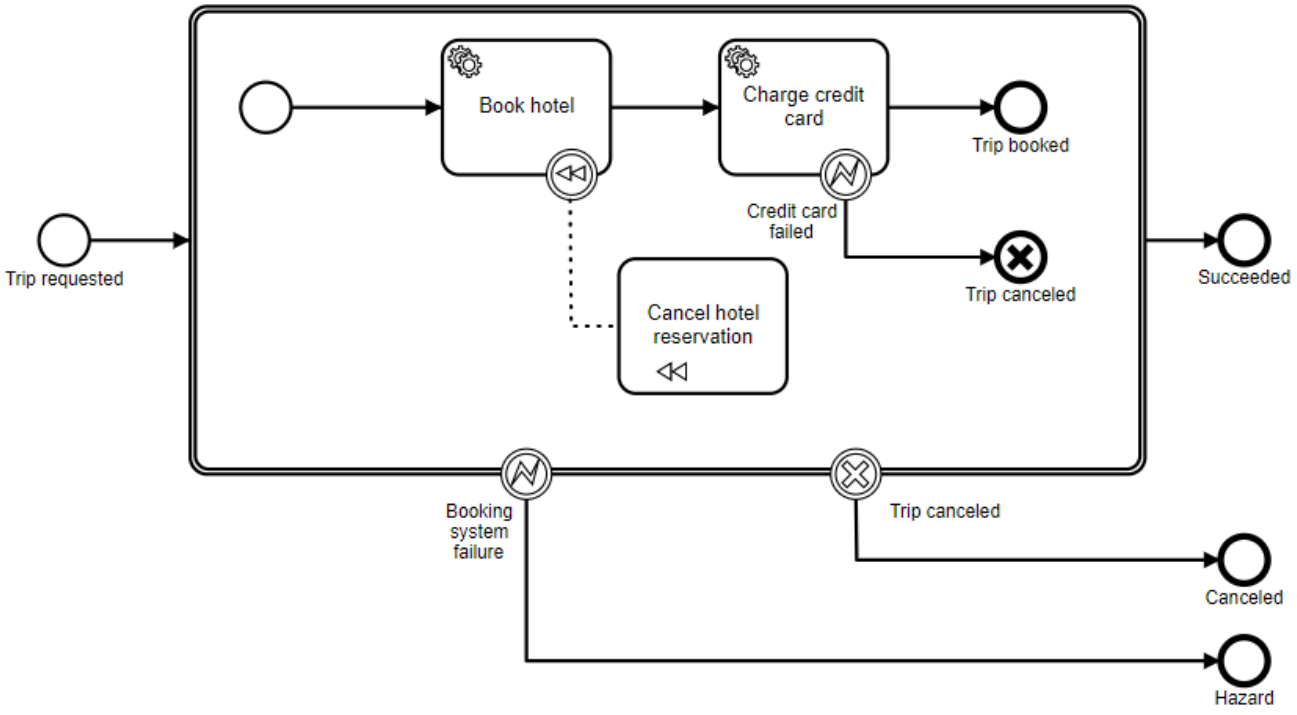

Рисунок 2.18 – Приклад BPMN-схеми з використовуванням підпроцесу транзакції

Завдання отримання (Receive Task) — це просте завдання, яке чекає на отримання певного повідомлення. Коли виконання процесу приходить до завдання отримання, стан процесу передається в сховище збереження. Це означає, що процес буде залишатися в цьому стані очікування, доки механізм не отримає конкретне повідомлення, яке ініціює продовження процесу за межами завдання отримання.

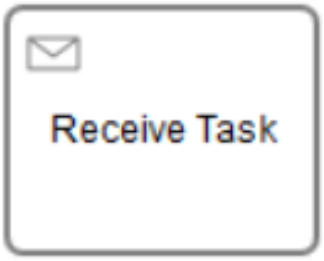

Рисунок 2.19 – Позначення завдання отримання на BPMN-схемі

Робота зі списком завдань

У наступному прикладі ми розглянемо типовий сценарій робочого процесу людини. У нашому готовому дистрибутиві Tasklist має чотирьох демонстраційних користувачів, які належать до різних груп користувачів. Увійдіть за допомогою демо-версії користувача.

Почніть процес

Щоб запустити екземпляр процесу за допомогою списку завдань, клацніть «Запустити процес» у меню заголовка інформаційної панелі та виберіть процес із відображеного списку визначень процесів. Якщо тут немає визначень процесів, переконайтеся, що ваша програма процесу розгорнута правильно та у вас є необхідні дозволи* для запуску процесу. Для нашого прикладу запустіть процес отримання рахунка-фактури. Вибравши процес для початку, заповніть форму початку. Натисніть «Пуск», щоб перейти до наступного кроку в нашому прикладі. Тепер на інформаційній панелі можна створити фільтр, який відображає це завдання. Для нашого прикладу ми створимо фільтр для відображення всіх завдань, призначених користувачеві, який увійшов у систему, і які належать до визначення процесу «Квитанція рахунка-фактури». Для цього натисніть «Створити фільтр» і вставте назву фільтра, наприклад, «Мої завдання рахунка-фактури». Потім натисніть «Критерії» та «Додати критерій». Далі клацніть порожнє поле «Ключ», виберіть

«Ім’я» у підменю «Визначення процесу» та вставте Квитанцію-фактуру в поле «Значення». Знову клацніть «Додати критерій», виберіть «Призначений» у підменю «Користувач/Група» ключового поля та вставте ${ currentUser() } у поле значення. Ми також хочемо опублікувати цей фільтр для наших колег. Для цього натисніть «Дозволи» та поставте прапорець «Доступно для всіх користувачів». На останньому етапі натисніть кнопку Зберегти, щоб створити фільтр. Тепер можна побачити фільтр у лівій частині приладової панелі.

Спочатку потрібно підтвердити це завдання, щоб мати можливість працювати над ним. Для цього клацніть на Претензію в розділі перегляду завдання. Ви також можете перепризначити завдання іншому користувачеві, натиснувши Demo Demo. З’явиться текстове поле, в яке можна вставити користувача, якому потрібно призначити це завдання. Після створення фільтра ми хочемо почати працювати над завданням. Ми можемо це зробити, вибравши завдання в результатах фільтрування. З правого боку ви побачите форму завдання, над якою потрібно працювати як вбудовану форму.

У нашому прикладі форми завдання вас просять підтвердити рахунок-фактуру (чи ні). Щоб виконати завдання, поставте прапорець біля пункту Чи схвалюєте ви? чи ні та натисніть Завершити. Для нашого прикладу встановіть прапорець і виконайте завдання. Потім подивіться на результати фільтра. Тепер ви побачите, що завдання Підготувати банківський переказ створено.

Коли ви надсилаєте форму завдання, завдання буде виконано, і процес продовжується в двигуні. Крім того, ви можете візуалізувати модель процесу, натиснувши вкладку «Діаграма» в розділі перегляду завдань на інформаційній панелі.

Завдання бізнес-правила використовується для синхронного виконання одного або кількох правил. Також можна викликати код Java або надати робочий елемент для зовнішнього працівника для асинхронного завершення або викликати логіку, яка реалізується у формі веб-сервісів.

Рисунок 2.20 – Позначення завдання бізнес правила на BPMN-схемі

Коли завдання бізнес-правила вирішує визначення рішення, яке підлягає оцінці, воно має враховувати багатопотоковість.

За замовчуванням ідентифікатор клієнта визначення викликаючого процесу використовується для оцінки визначення рішення. Тобто, якщо визначення процесу, що викликає, не має ідентифікатора клієнта, тоді завдання бізнес-правила оцінює визначення рішення за допомогою наданого ключа, прив’язуваного та без ідентифікатора клієнта. Якщо визначення викликаючого процесу має ідентифікатор клієнта, оцінюється визначення рішення з наданим ключем і тим самим ідентифікатором клієнта.

Зауважте, що ідентифікатор клієнта екземпляра викликаючого процесу не враховується в поведінці за замовчуванням.

Завдання скрипт — це автоматизована діяльність. Коли виконання процесу надходить до завдання скрипта, виконується відповідний скрипт

Значенням атрибута формата скрипта має бути ім’я, сумісне з JSR-223 (Скрипти для платформи Java).

Вихідний код сценарію необхідно додати як текстовий вміст дочірнього елемента скрипта. Крім того, вихідний код можна вказати як вираз або зовнішній ресурс.

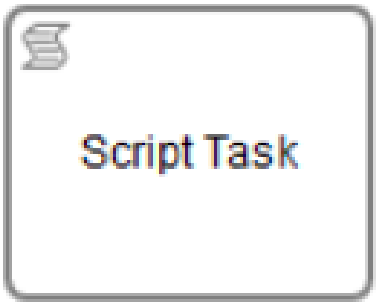

Рисунок 2.21 – Позначення завдання скрипта на BPMN-схемі

Завдання «вручну» визначає завдання, яке є зовнішнім для двигуна BPM. Він використовується для моделювання роботи, яка виконується кимось, кого двигун не повинен знати, і який не має відомої системи чи інтерфейсу інтерфейсу. Для двигуна завдання вручну обробляється як перехідна діяльність, автоматично продовжуючи процес, коли виконання процесу досягає його.

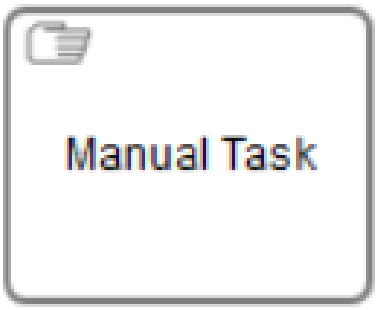

Рисунок 2.22 – Позначення завдання скрипта на BPMN-схемі

Також до різних типів завдань, ми можемо позначати завдання як цикли, кілька екземплярів або компенсацій. Маркери можна комбінувати з типами завдань.

Багатоекземплярність або виконання із багатьма екземплярами – це спосіб визначення повторення певного кроку бізнес-процесу. У концепціях програмування мультиекземпляр відповідає кожній конструкції: він дозволяє виконувати певний крок або навіть повний підпроцес для кожного елемента в даній колекції, послідовно або паралельно.

Мультиекземпляр — це звичайна активність або завдання, яка має додаткові властивості (так звані характеристики кількох екземплярів), які призведуть до виконання дії кілька разів під час виконання. Наступні дії можуть стати діяльністю з кількома екземплярами:

- Сервісне завдання

- Надіслати завдання
- Завдання користувача
- Завдання з правилами бізнесу
- Скриптове завдання
- Отримати завдання
- Ручне завдання
- (Вбудований) Підпроцес
- Діяльність дзвінків
- Підпроцес транзакції
- Шлюз або подія не можуть стати кількома примірниками.

Якщо дія є мультиекземпляром, це позначається трьома короткими лініями внизу завдання. Три вертикальні лінії вказують на те, що екземпляри будуть виконуватися паралельно, а три горизонтальні лінії вказують на послідовне виконання.

Задача користувача MI (паралельна)

Сервісна задача MI (послідовна)

Відповідно до вимог специфікації, кожне батьківське виконання створених виконання для кожного екземпляра матиме такі змінні:

Крім того, кожне зі створених виконання матиме локальні змінні виконання (тобто, не видиму для інших екземплярів та не зберігатиме змінні на рівні екземпляра процесу):

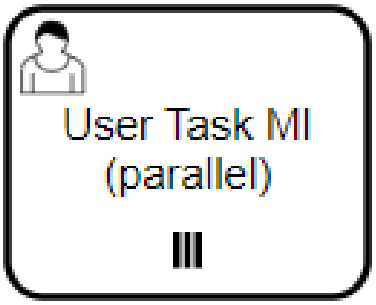

Рисунок 2.23 – Позначення мультиекземляра завдання користувача на BPMN-схемі

Рисунок 2.24 – Позначення мультиекземляра сервісного завдання на BPMN-схемі

Події таймера – це події, які ініціюються визначеним таймером. Їх можна використовувати як стартову подію, проміжну подію або граничну подію. Граничні події можуть бути перериваючими чи ні.

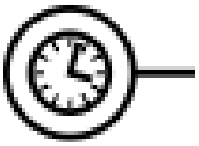

Рисунок 2.25 – Позначення події таймера на BPMN-схемі

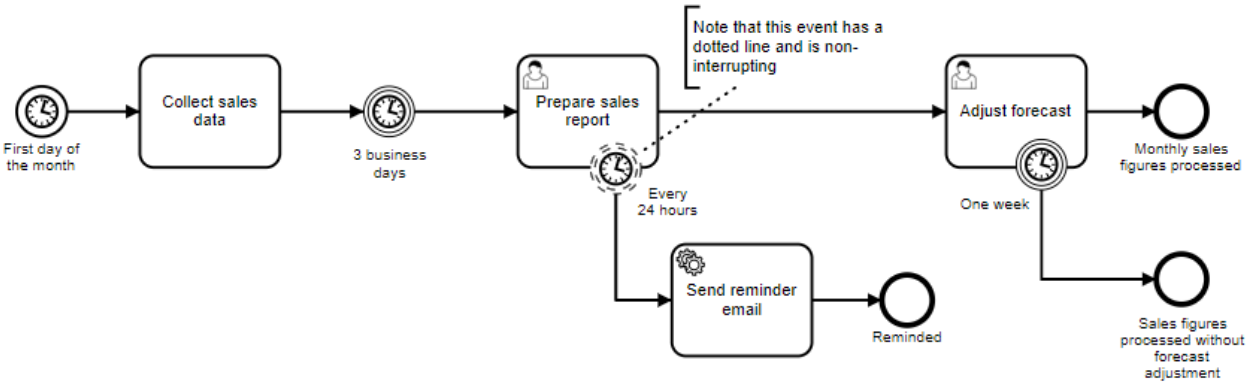

Рисунок 2.26 – Приклад BPMN-схеми з використанням подій-таймерів

Таймери налаштовуються за допомогою формату часу ISO 8601. Визначення таймера має містити точно один із наступних елементів. Щоб вказати, як довго таймер повинен працювати перед його запуском, timeDuration можна вказати як

піделемент timerEventDefinition. Можна визначити тривалість у двох різних форматах тривалості ISO 8601.

Визначає повторювані інтервали, які можуть бути корисними для періодичного запуску процесу або для надсилання кількох нагадувань про прострочені завдання користувача. Елемент циклу часу може мати два формати. Одним з варіантів є формат тривалості повторюваного часу, як зазначено у стандарті повторюваних інтервалів ISO 8601. Крім того, можна вказати часовий цикл за допомогою виразів cron, приклад нижче показує, що тригер спрацьовує кожні 5 хвилин, починаючи з повної години. Параметр тривалості повторюваного часу краще підходить для обробки відносних таймерів, які обчислюються відносно певного моменту часу (наприклад, часу, коли було запущено завдання користувача), тоді як вирази cron можуть обробляти абсолютні таймери, що особливо корисно для події запуску таймера. Циклом повторення таймера можна керувати за допомогою REST API або викликом ManagementService. Встановивши термін виконання таймера, можна змінити момент часу, коли таймер виконується. Зміни в одному екземплярі таймера не впливають автоматично на наступні екземпляри таймера. Наприклад, певний повторюваний таймер створює подію таймера кожні 30 хвилин. Якщо дату виконання однієї події таймера змінено (наприклад, +15 хвилин), вона буде виконана через 45 хвилин після попереднього таймера. Однак наступний таймер буде виконувати старий шаблон і виконуватиметься через 15 хвилин після зміненого таймера. Ви можете використовувати вирази для визначення подій таймера. Таким чином ви можете впливати на визначення таймера на основі змінних процесу. Змінні процесу повинні містити рядок ISO 8601 (або cron для типу циклу) для відповідного типу таймера.

Подія запуску таймера використовується для створення екземпляра процесу в певний час. Його можна використовувати як для процесів, які мають запускатися лише один раз, так і для процесів, які мають запускатися через певні проміжки часу.

Проміжна подія таймера діє як секундомір. Коли виконується перехоплення подій, запускається таймер. Коли таймер спрацьовує (наприклад, через певний інтервал), виконується потік послідовності, що виходить із проміжної події таймера.

Гранична подія таймера діє як секундомір і як будильник. Коли настає виконання в дії, до якої приєднано граничну подію, запускається таймер. Коли таймер спрацьовує (наприклад, через певний інтервал), діяльність переривається, і виконується потік послідовності, що виходить за межу події таймера.

Існує різниця між подією таймера, яка перериває і не перериває. За замовчуванням є подія, що перериває. Подія без переривання призводить до того, що початкова діяльність не переривається, а діяльність залишається там.

Помилки – це події, які ініціюються певною помилкою.

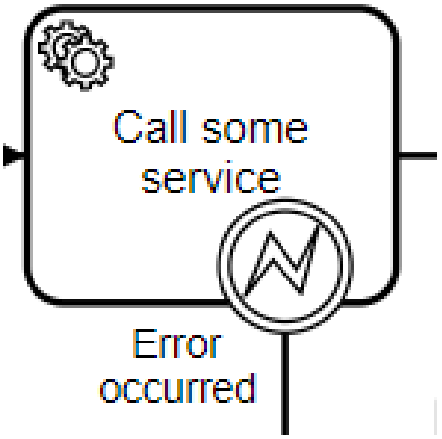

Рисунок 2.27 – Позначення події помилки на BPMN-схемі

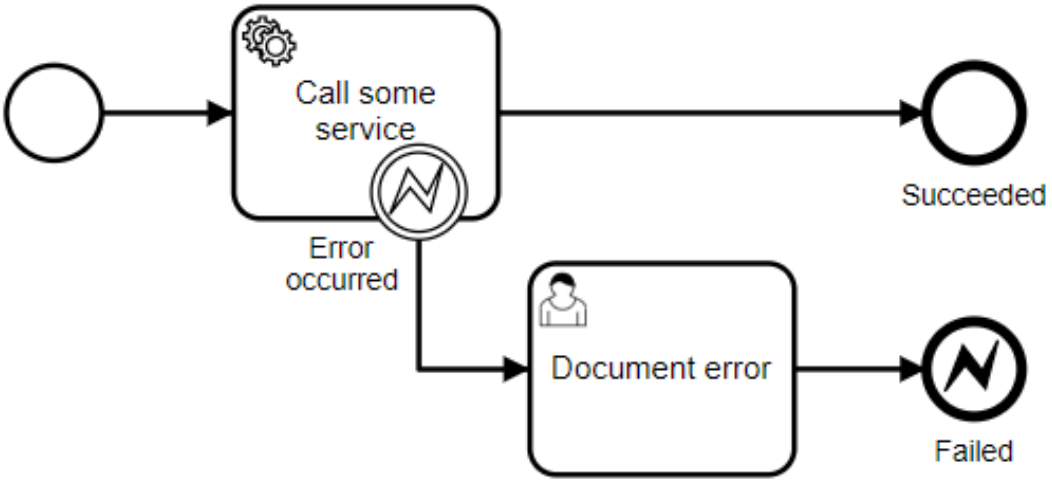

Рисунок 2.28 – Приклад BPMN-схеми з використанням подій помилки

Помилка BPMN призначена для ділових помилок, які відрізняються від технічних винятків. Отже, це відрізняється від винятків Java, які за замовчуванням обробляються по-своєму. Визначення події помилки посилається на елемент помилки.

Події ескалації – це події, які посилаються на іменовану ескалацію. Здебільшого вони використовуються для зв'язку від підпроцесу до вищого процесу. На відміну від помилки, подія ескалації не є критичною, і виконання продовжується в місці викиду.

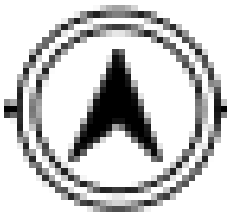

Рисунок 2.29 – Позначення події ескалації на BPMN-схемі

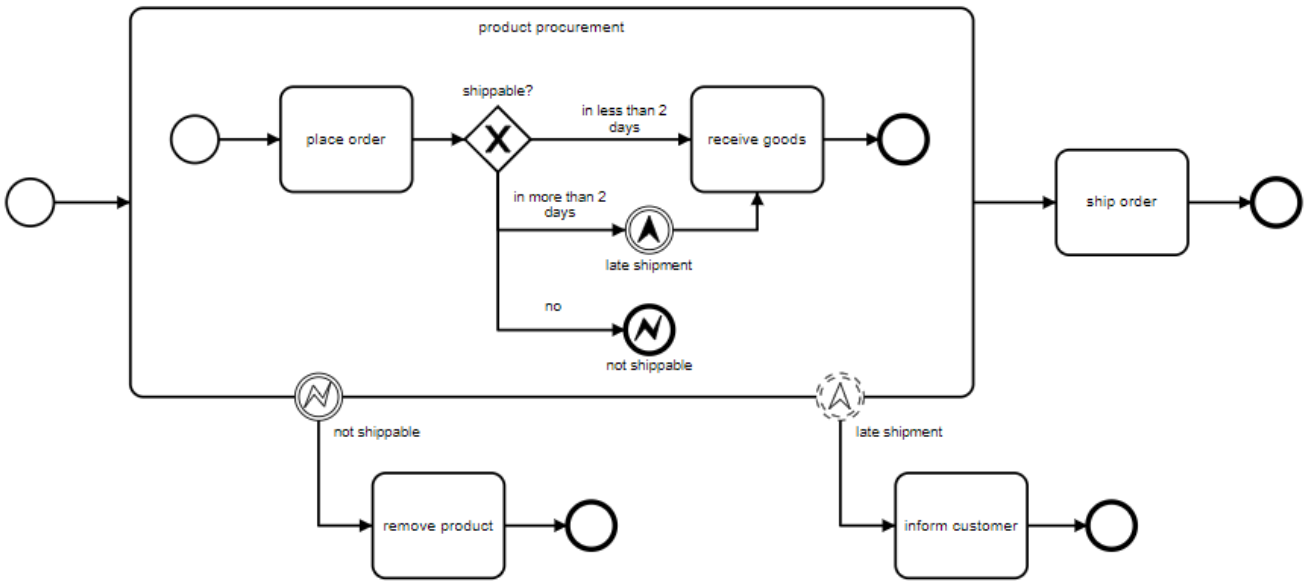

Рисунок 2.30 – Приклад BPMN-схеми з використанням подій ескалацій

Визначення події ескалації оголошується за допомогою елемента escalationEventDefinition. Атрибут escalationRef посилається на елемент ескалації,

оголошений як дочірній елемент кореневого елемента визначення. Подію початку ескалації можна використовувати лише для ініціювання підпроцесу події - її не можна використовувати для запуску екземпляра процесу. Підпроцес події з подією початку ескалації ініціюється подією ескалації, яка відбувається в тій самій або в нижчому діапазоні (наприклад, підпроцес або активність виклику). Коли підпроцес ініціюється подією ескалації від дії виклику, тоді визначені вихідні змінні дії виклику передаються до підпроцесу.

Сигнальні події – це події, які посилаються на іменований сигнал. Сигнал є подією глобального масштабу (семантика широкомовної передачі) і доставляється всім активним обробникам.

Нижче наведено приклад двох окремих процесів, які спілкуються за допомогою сигналів. Перший процес починається, якщо страховий поліс оновлено або змінено.

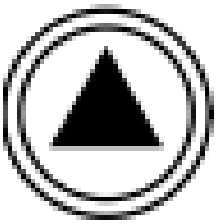

Рисунок 2.31 – Позначення сигнальної події на BPMN-схемі

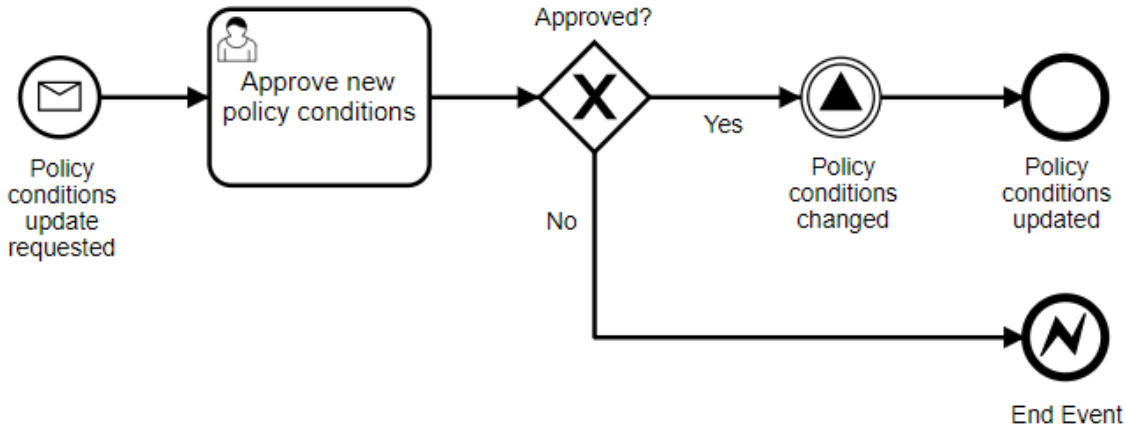

Рисунок 2.32 – Приклад BPMN-схеми з використанням сигнальної події

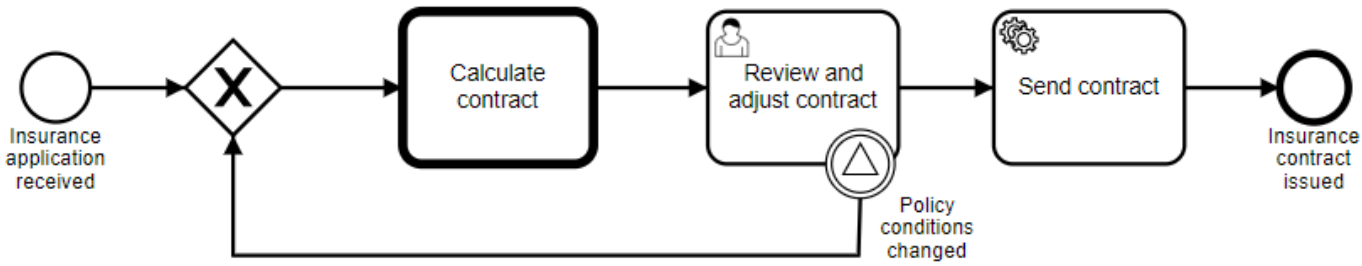

Рисунок 2.33 – Приклад BPMN-схеми з використанням сигнальної події прив’язаної до конкретного завдання користувача

Ви можете використовувати вирази для імені у визначенні сигнальної події. Ім’я потім вирішується, як тільки процес досягає області дії сигналу. Наприклад, коли екземпляр процесу досягає проміжної події перехоплення сигналу, тоді вираз у імені розв’язується.

Використовуючи вирази в імені сигналу, ви можете динамічно впливати на назву сигналу на основі змінних процесу. Це може стати в нагоді, якщо, наприклад, є необхідність переривати паралельні розгалуження. Сигнал може передаватися екземпляром процесу за допомогою конструкції BPMN або програмно за допомогою Java API. Якщо вказано ідентифікатор виконання, сигнал доставляється лише до конкретного виконання. В іншому випадку сигнал глобально передається всім передплатним обробникам (семантика широкомовної передачі).

Примітка. Сигнальна подія не виконує жодної кореляції з конкретним екземпляром процесу. Навпаки, він транслюється на всі екземпляри процесу. Якщо вам потрібно виключно доставити сигнал до конкретного екземпляра процесу, не використовуйте подію сигналу викиду. Замість цього виконайте кореляцію вручну, використовуючи відповідні механізми запитів, і доставте сигнал до певного виконання програмно.

Подія початку сигналу може бути використана для запуску екземпляра процесу за допомогою іменованого сигналу. Під час розгортання визначення

процесу з одним або кількома подіями початку сигналу застосовуються такі міркування:

Ім’я події початку сигналу має бути унікальним для даного визначення процесу, тобто визначення процесу не повинно містити кілька подій запуску сигналу з однаковою назвою. Механізм генерує виняток після розгортання визначення процесу, якщо два або більше подій початку сигналу посилаються на той самий сигнал або якщо два або більше події початку сигналу посилаються на сигнали з однаковою назвою сигналу.

На відміну від подій початку повідомлення, ім’я події початку сигналу не обов’язково має бути унікальним для всіх визначень розгорнутих процесів.

Управління версіями процесу: після розгортання нової версії визначення процесу передплати на сигнал попередньої версії скасовуються. Це також стосується сигнальних подій, яких немає в новій версії.

Примірник процесу визначення процесу з одним або кількома подіями початку сигналу буде запущено, коли буде викинутий сигнал з власною назвою.

Коли відбувається виконання дії, до якої приєднана подія межі сигналу, подія межі сигналу перехоплює сигнали з власною назвою.

Примітка: на відміну від інших подій, таких як подія межі помилки, подія межі сигналу не лише ловить сигнальні події, що видаються з області, до якої вона приєднана. Сигнальна подія має глобальну область (семантику широкомовної передачі), що означає, що сигнал може передаватися з будь-якого місця, навіть з іншого екземпляра процесу.

Подія завершення скасування може використовуватися лише в поєднанні з підпроцесом транзакції. Коли досягається подія завершення скасування, створюється подія скасування, яка повинна бути перехоплена граничною подією скасування. Гранична подія скасування скасовує транзакцію та ініціює компенсацію.

Приєднана проміжна подія скасування перехоплення на межі підпроцесу транзакції, або, коротко, гранична подія скасування, ініціюється, коли транзакцію скасовується. Коли ініціюється гранична подія скасування, вона спочатку перериває всі активні виконання в поточній області. Далі він починає компенсацію всіх

активних граничних подій компенсації в рамках транзакції. Компенсація виконується синхронно, тобто гранична подія чекає до завершення компенсації, перш ніж вийти з транзакції. Коли компенсація завершена, підпроцес транзакції залишається з використанням потоку(ів) послідовності, що закінчується граничною подією скасування. Компенсація запуску: Компенсація може бути запущена для визначеної діяльності або для області, в якій розміщена подія компенсації. Компенсація виконується через виконання обробника компенсації, пов'язаного з діяльністю.

Коли для дії генерується компенсація, пов'язаний обробник компенсації виконується стільки ж разів, скільки успішно завершено дію.

Якщо виплачується компенсація для поточної області, компенсуються всі дії в поточній області, що включає дії в одночасних гілках.

Компенсація запускається ієрархічно: якщо діяльність, яка підлягає компенсації, є підпроцесом, компенсація запускається для всіх дій, що містяться в підпроцесі. Якщо підпроцес містить вкладені дії, компенсація викидається рекурсивно. Однак компенсація не поширюється на «верхні рівні» процесу: якщо компенсація запускається всередині підпроцесу, вона не поширюється на дії за межами підпроцесу. Специфікація BPMN стверджує, що компенсація ініціюється за дії на «тому ж рівні підпроцесу».

Компенсація споживається підпроцесом події компенсації: якщо діяльність, яка підлягає компенсації, є підпроцесом, а підпроцес містить підпроцес подій, ініційований подією початку компенсації, компенсація запускає підпроцес події замість того, щоб ініціювати дії, що містяться в підпроцесі.

Компенсація здійснюється в порядку, зворотному виконанню. Це означає, що будь-яка діяльність, виконана останньою, отримує компенсацію першою тощо.

Подія проміжної компенсації викиду може використовуватися для компенсації підпроцесів транзакцій, які успішно завершені.

Примітка. Якщо компенсація генерується в межах області, яка містить підпроцес, а підпроцес містить дії з обробниками компенсації, компенсація поширюється на підпроцес лише в тому випадку, якщо вона успішно завершилася

під час виклику компенсації. Якщо деякі дії, вкладені всередині підпроцесу, завершилися і до них приєднано обробники компенсації, обробники компенсації не виконуються, якщо підпроцес, що містить ці дії, ще не завершено.

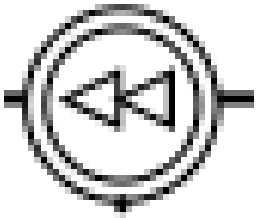

Рисунок 2.34 – Позначення події компенасції на BPMN-схемі

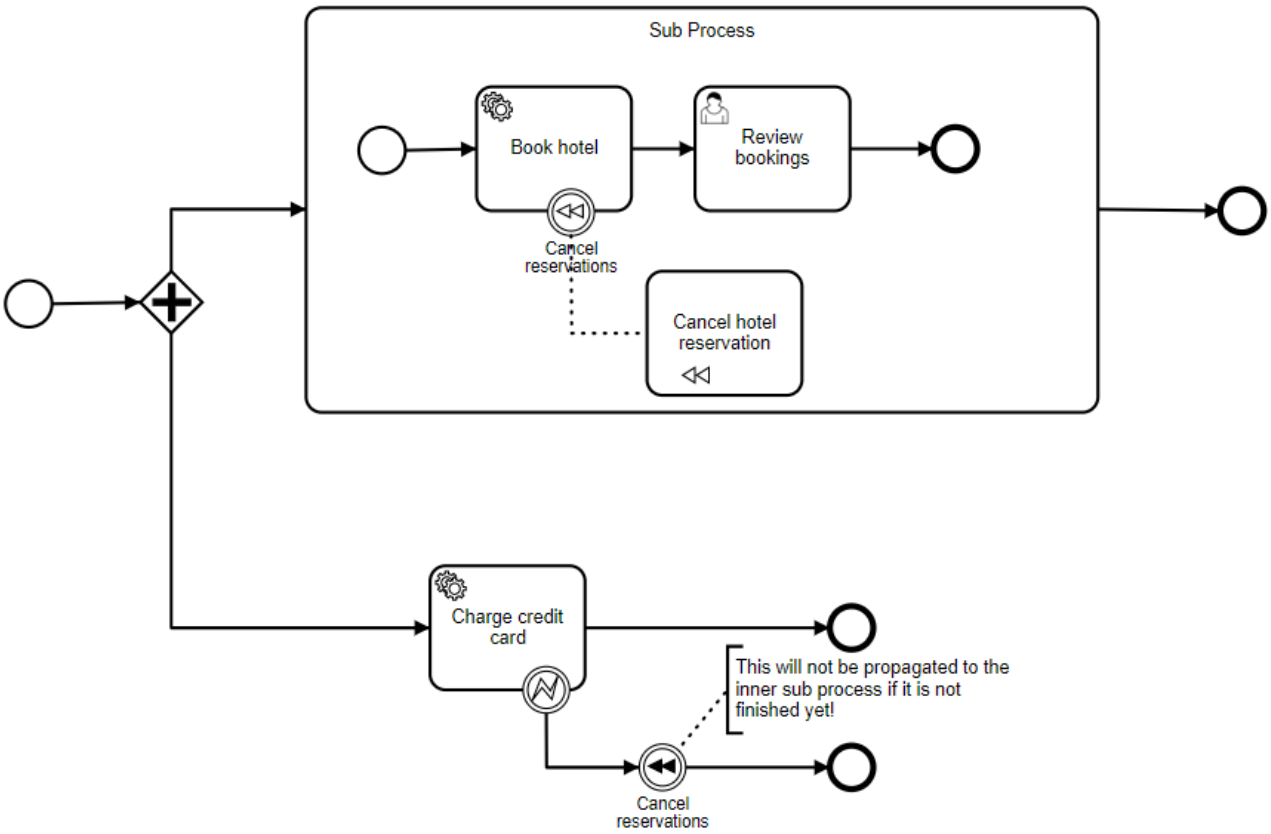

Рисунок 2.35 – Приклад BPMN-схеми з використанням події компенсації

У цьому процесі ми маємо два одночасних виконання: одне виконує вбудований підпроцес, а друге — операцію «стягнення кредитної картки». Припустимо, що обидва виконання запущено, і перше одночасне виконання очікує, поки користувач виконає завдання «перегляд бронювання». Друге виконання виконує операцію «стягнення плати з кредитної картки», і виникає помилка, яка спричиняє подію «скасувати бронювання», щоб ініціювати компенсацію. На цьому етапі паралельний підпроцес ще не завершено, що означає, що подія компенсації не поширюється на підпроцес, і, таким чином, обробник компенсації «скасувати бронювання готелю» не виконується. Якщо завдання користувача (і, таким чином, вбудований підпроцес) завершується до виконання «скасування резервування», компенсація поширюється на вбудований підпроцес.

Примітка: Коли компенсація генерується для дії з кількома екземплярами, пов’язаний обробник компенсації виконується лише тоді, коли всі екземпляри цієї дії закінчуються. Це означає, що діяльність кількох екземплярів має бути закінчена, перш ніж її можна буде компенсувати.

Змінні процесу: під час компенсації вбудованого підпроцесу виконання, яке використовується для виконання обробників компенсації, має доступ до локальних змінних процесу підпроцесу в тому стані, в якому вони були, коли підпроцес завершив виконання. Щоб досягти цього, робиться знімок змінних процесу, пов’язаних із виконанням області (виконання, створене для виконання підпроцесу). З цього випливає кілька наслідків:

Обробник компенсації не має доступу до змінних, доданих до одночасних виконання, створених у межах підпроцесу.

Змінні процесу, пов’язані з виконанням вище в ієрархії, наприклад, змінні процесу, пов’язані з виконанням екземпляра процесу, не містяться в моментальному знімку: обробник компенсації має доступ до цих змінних процесу в стані, в якому вони перебувають під час виклику компенсації.

Змінний знімок робиться лише для вбудованих підпроцесів, а не для інших дій.

Поточні обмеження: сама компенсація в даний час здійснюється за допомогою одночасних страт. Одночасні виконання розпочинаються в порядку, зворотному тому, до якого завершилися компенсовані дії. Майбутні версії діяльності можуть включати можливість послідовного виконання компенсації.

Компенсація не поширюється на екземпляри підпроцесу, породжені діяльністю виклику.

Події посилання є окремим випадком - воно не має спеціальної семантики виконання, але служить «Перейти» до іншої точки в тій же моделі процесу (точніше: в тому ж підпроцесі). Отже, ви можете використовувати два збігаються посилання як альтернативу потоку послідовності, як показано в наступному прикладі.

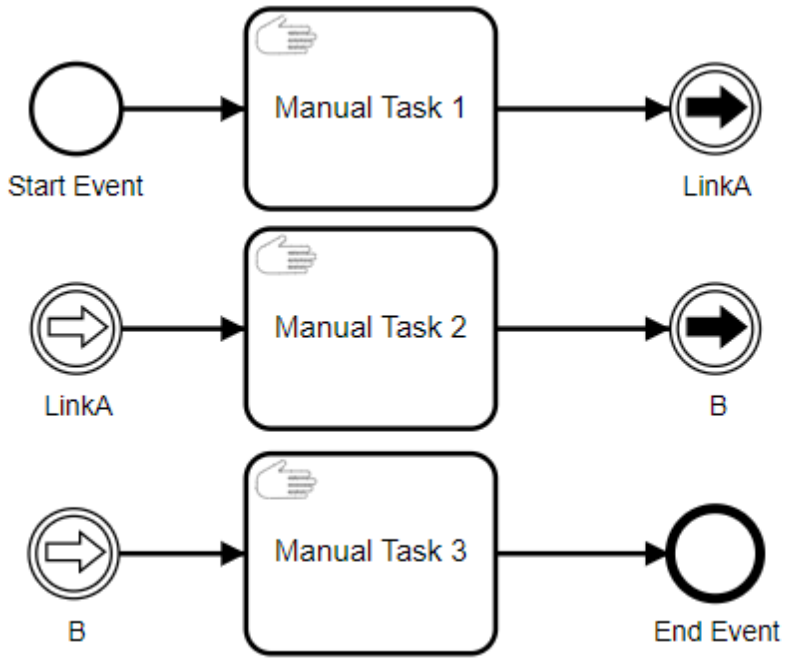

Рисунок 2.36 – Приклад BPMN-схеми з використанням події посилання

Подія екстренного завершення завершує повну область, в якій вона викликана, і всі внутрішні області. Це корисно, якщо є паралельний потік маркерів у процесі, і необхідно негайно використати всі маркери, доступні в одній області.

Подія завершення на рівні екземпляра процесу завершує весь екземпляр процесу. На рівні підпроцесу поточна область і всі наявні екземпляри процесів будуть припинені.

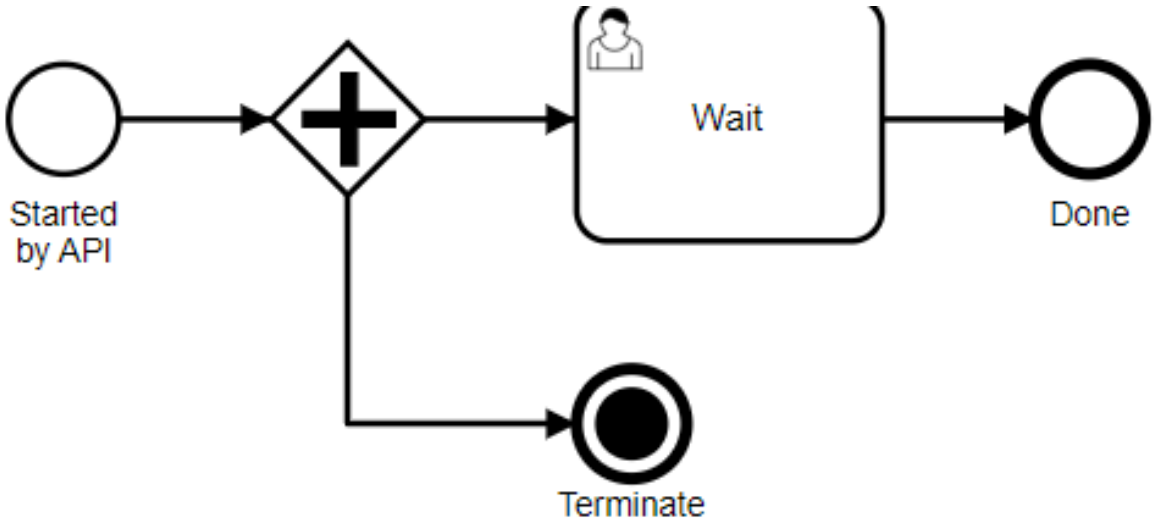

Рисунок 2.37 – Приклад BPMN-схеми з використанням події екстренного завершення

Рисунок 2.38 – Позначення події екстренного завершення на BPMN-схемі

Шлюз на основі подій дозволяє приймати рішення на основі подій. Кожен вихідний потік послідовності шлюзу повинен бути підключений до проміжної події перехоплення. Коли виконання процесу досягає шлюзу на основі подій, шлюз діє як стан очікування: виконання призупинено. Крім того, для кожного вихідного потоку послідовності створюється підписка на подію.

Зауважте, що потоки послідовності, що закінчуються шлюзом на основі подій, відрізняються від звичайних потоків послідовностей. Ці потоки послідовності ніколи насправді не «виконуються». Навпаки, вони дозволяють механізму процесу визначати, на які події має підписатися виконання, що надходить на шлюз на основі подій.

Наступний процес є прикладом процесу зі шлюзом на основі подій. Коли виконання надходить до шлюзу на основі подій, виконання процесу призупиняється.

Крім того, екземпляр процесу підписується на подію сигналу попередження і створює таймер, який запускається через 10 хвилин. Це фактично змушує процесор чекати десять хвилин на подію сигналу. Якщо подія сигналу відбувається протягом 10 хвилин, таймер скасовується, а виконання продовжується після сигналу. Якщо сигнал не спрацьовує, виконання продовжується після таймера, а підписка на сигнал скасовується.

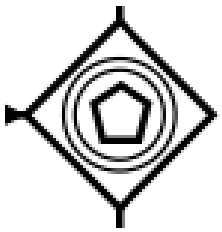

Рисунок 2.39 – Позначення шлюзу на основі подій на BPMN-схемі

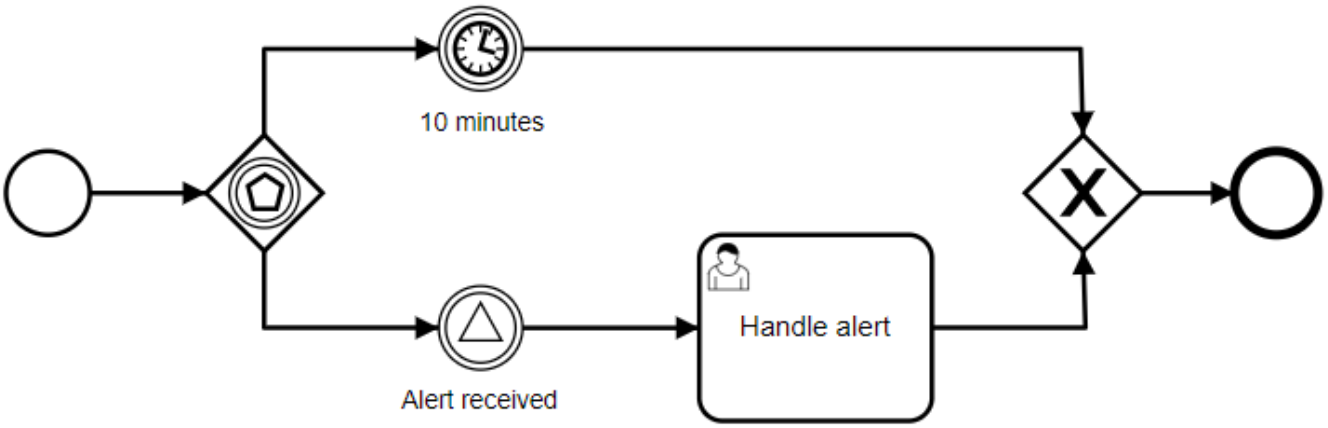

Рисунок 2.40 – Приклад BPMN-схеми з використанням шлюзу на основі подій

2.3 Основні можливості інструмента Task List

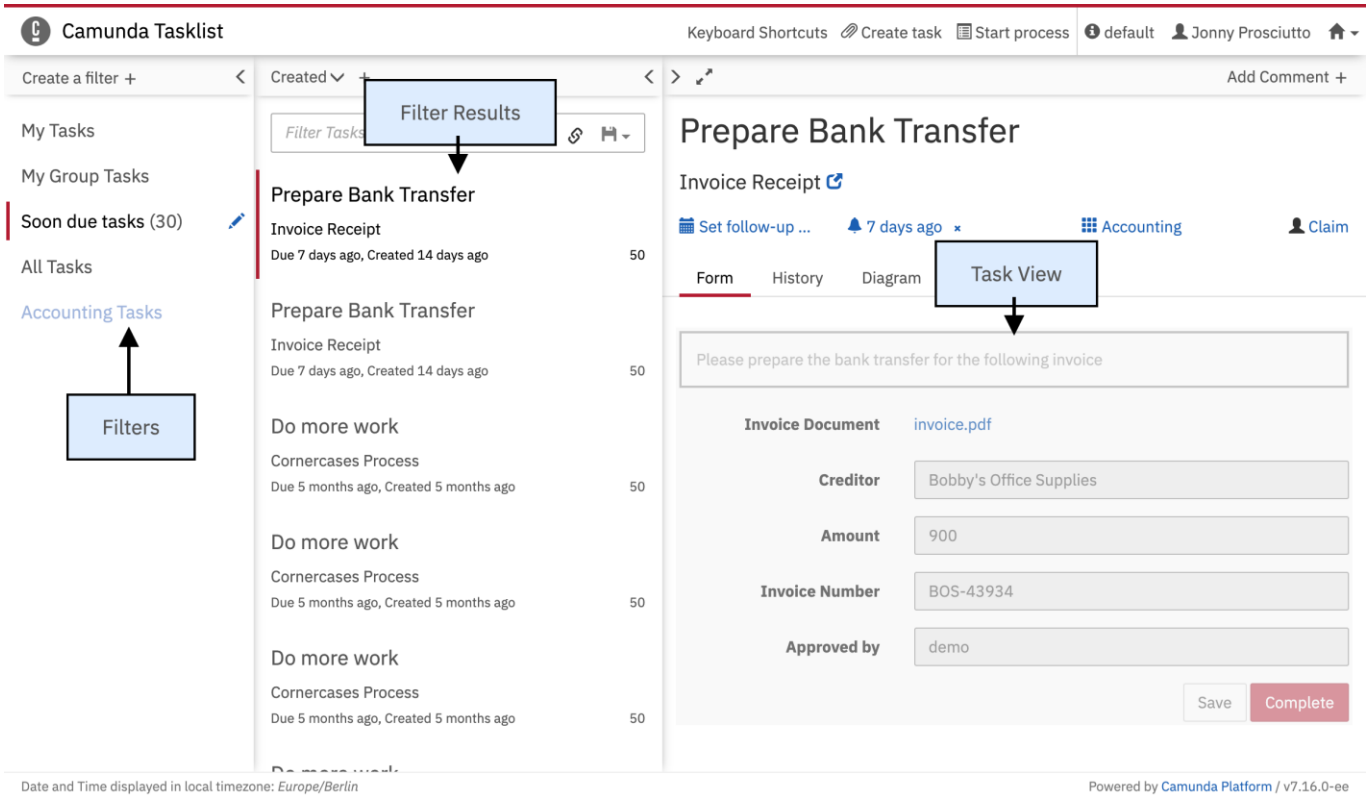

Рисунок 2.41– Загальний вигляд Camunda Task List з відмітками основних частин

На інформаційній панелі списку завдань ви бачите огляд незавершених завдань. У лівій частині екрана відображається огляд фільтрів. У верхньому правому куті екрана ви можете встановити дату виконання або термін виконання, ви можете заявити, скасувати витребування та перепризначити завдання, а також додавати коментарі. Під цим розділом відображається вбудована форма (зверніть увагу, що зовнішні форми завдань тут не відображаються), ви можете перейти до історії завдань, побачити діаграму або переглянути опис завдання користувача. У списку завдань ви можете створювати та вибирати фільтри. Ви можете використовувати ці фільтри для створення списків завдань, відсортованих за заданими критеріями. Щоб створити фільтр, виберіть Створити фільтр. Після цього ви побачите екран, як показано на зображенні вище. У вас є кілька варіантів налаштування фільтра: Загальні – вкажіть назву та опис фільтра, а також призначте колір. Призначте пріоритет, щоб визначити порядок відображення фільтрів на інформаційній панелі. Ви можете вибрати, щоб фільтр автоматично оновлював результати фільтра,

встановивши прапорець Автоматичне оновлення. Інтервал оновлення за замовчуванням становить 10 секунд. Дозволи – вкажіть, які користувачі або групи можуть бачити фільтр. Ви можете встановити фільтр як глобально доступний, встановивши прапорець Доступний для всіх користувачів. Дозвіл, встановлений тут, еквівалентний дозволу READ, який також можна встановити в Camunda Admin. Якщо ви хочете призначити інші дозволи, ви можете зробити це на вкладці «Авторизації» в адміністраторі Camunda. Критерії – вкажіть, які завдання будуть відображатися при виборі фільтра. Необхідно вставити ключ і значення. Існують різні ключі, які можна вибрати з категорій Примірник процесу (ідентифікатор, бізнес-ключ), Визначення процесу (ідентифікатор, ключ, ім'я), екземпляр справи (ідентифікатор, бізнес-ключ), Визначення випадку (ідентифікатор, ключ, ім'я), інше (Стан екземпляра процесу, ідентифікатор екземпляра діяльності, ідентифікатор виконання), користувач/група (отримувач, власник, кандидат-користувач або група, залучений користувач, не призначений, стан делегування), завдання (ключ визначення, ім'я, опис, пріоритет) і дати (створення дата, дата виконання, дата подальшого виконання). Клавіші, позначені символом *, приймають вирази як значення. Змінні – вкажіть, які змінні відображатимуться в розділі результатів фільтрування на інформаційній панелі. Встановлення змінних тут не впливає на те, які завдання відображаються. Щоб встановити змінні, вам потрібно вставити ім’я, яке є закодованим ім’ям змінної, і мітку, яка визначає назву змінної в результатах фільтрування.

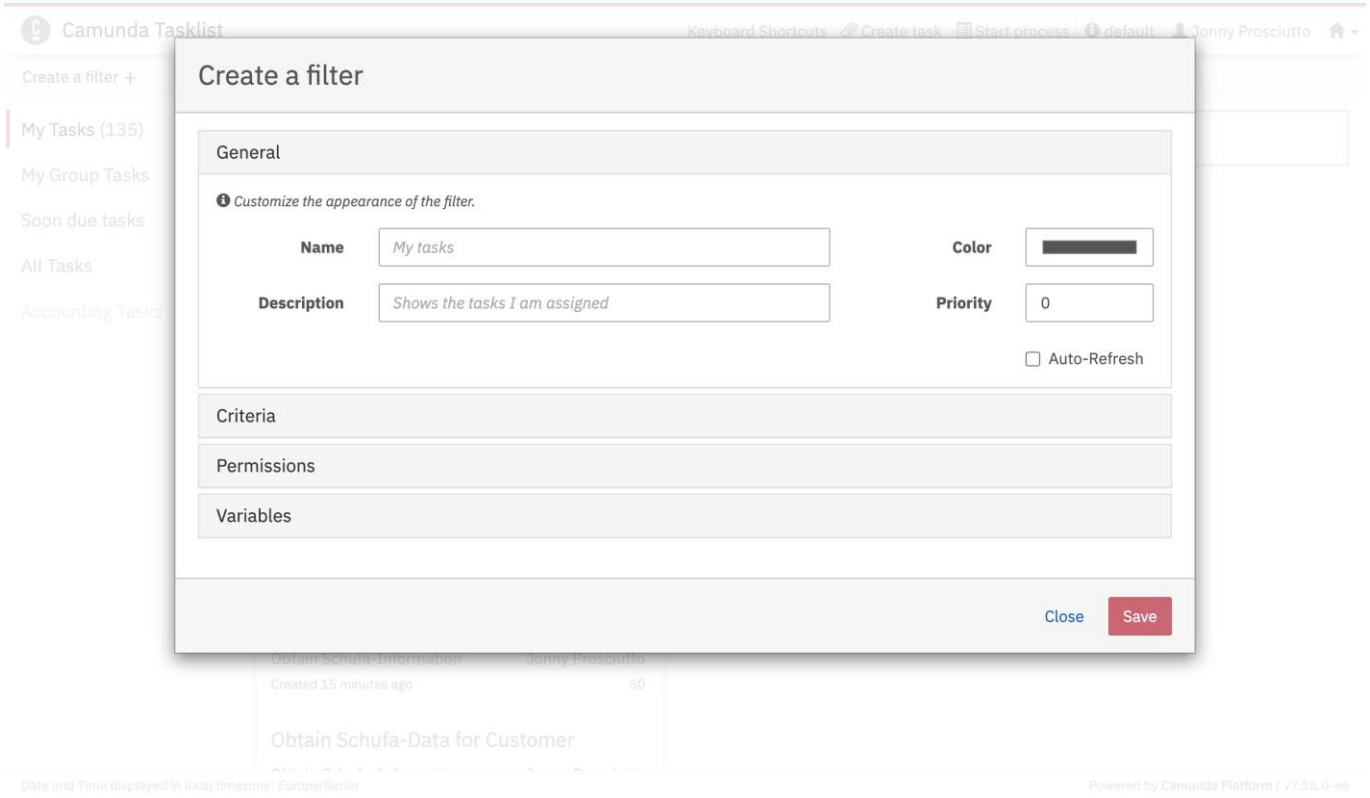

Рисунок 2.42 – Створення нового фільтра

Camunda Platform Cockpit — це веб-додаток для моніторингу та операцій. Він надає доступ до розгорнутих процесів BPMN і рішень DMN, дозволяє шукати запущені та завершені екземпляри та виконувати операції з ними.

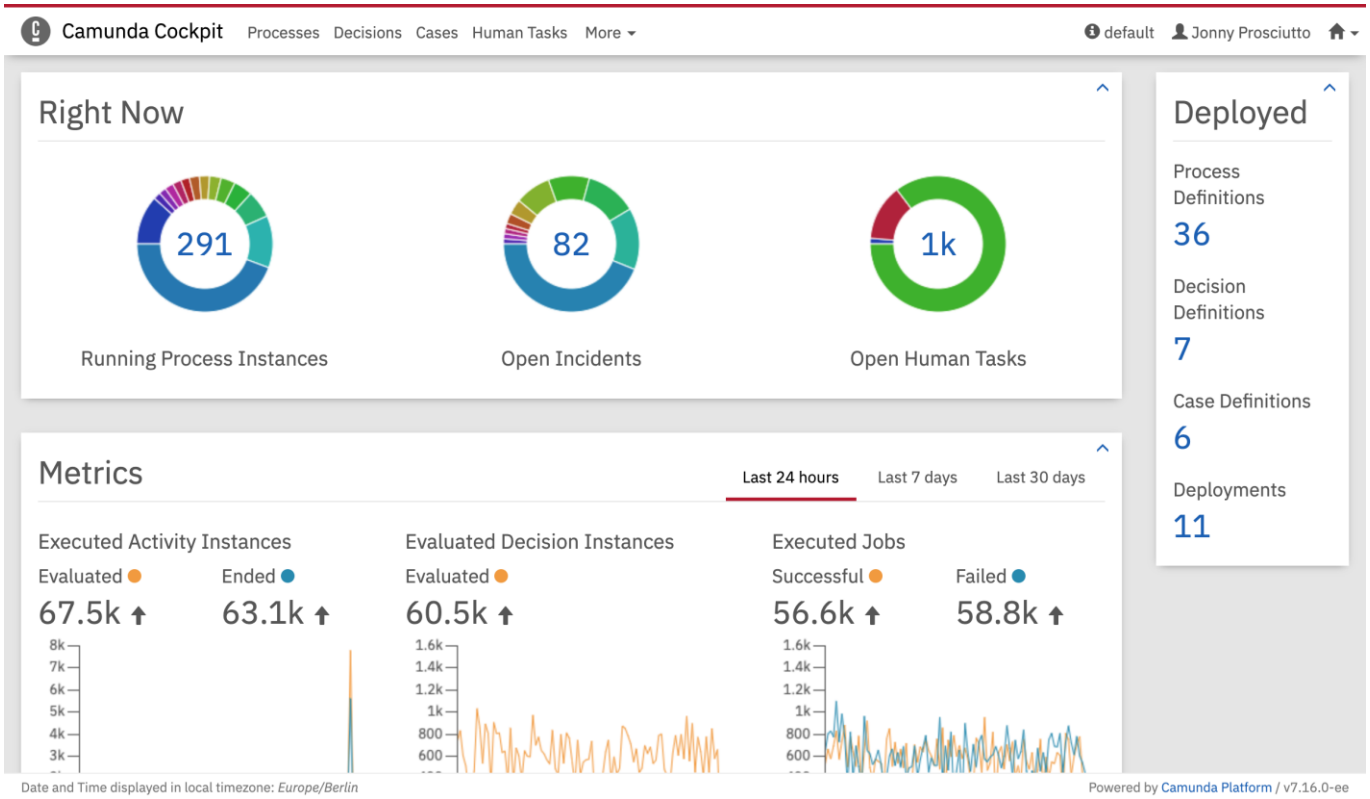

Рисунок 2.43 – Загальний вигляд Camunda **Cockpit**

2.6 Можливості REST-api Camunda

Серед усього іншого для покращення управління та підтримки бізнес-процесів у вигляді веб-сервісів у Camunda представлено стандартне Rest-api, що надає можливості маніпуляції з процесами під час їх виконання. Далі будуть представлені основні функції, які знадобляться далі при звертанні до процеса та у випадку залучення фронт-енд розробника чи тестувальника програмних систем.

Результатом REST API є доступ до доступу до всіх важливих сторінок.

Метою REST API є надання доступу до всіх відповідних інтерфейсів движка. Описані методи працюють на механізмі процесу за замовчуванням, який надається доступною службою ProcessEngineProvider. Можна додати /engine/{name} до будь-якого з HTTP-методів (якщо інше не задокументовано) для доступу до іншого механізму, де {name} — це назва механізму процесу, який повертається ProcessEngine#getName(), наприклад, /engine/myEngineName/ завдання. Для кожного методу ця документація містить можливі коди статусу HTTP. Пояснення коду помилки не охоплюють усі можливі причини помилок, які можуть виникнути під час обслуговування запиту, наприклад, більшість запитів не працюватиме належним чином, якщо є проблеми з доступом до бази даних. Будь-яка з цих недокументованих помилок буде перекладено на помилку HTTP 500.

2.5 Вибір технологій для розробки веб-додатку

В якості технологій для розробки веб-додатку було використано стек-технологій або так званий фреймворк, Spring. А саме його компоненти SpringBoot. В якості системи управління базами даних обрано PostgreSQL.

Через громіздку конфігурацію залежностей настроювання Spring для корпоративних додатків перетворилося на вельми стомлююче і схильне до помилок заняття. Особливо це стосується додатків, які використовують також кілька сторонніх бібліотек

Створюючи корпоративний Java-додаток на основі Spring, необхідно зробити деякі кроки для налаштування:

Залежно від використаних частин стеку-технологій (Spring MVC, Spring JDBC, Spring ORM тощо) імпортувати необхідні Spring-модулі.

Далі потрібно бібліотеку web-контейнерів. Імпортувати необхідні сторонні бібліотеки (Hibernate, Jackson).

Конфігурувати компоненти DAO, такі як: джерела даних, управління транзакціями і т.д.

Конфігурувати компоненти web-шару, такі як: диспетчер ресурсів, view resolver

Визначити клас, який завантажить усі необхідні конфігурації

Spring Boot - це проект за мету в якого стоїть полегшення та спрощення створення програм на основі Spring. Він дозволяє найпростішим способом створити web-додаток, вимагаючи від розробників мінімум зусиль з його настроювання та написання коду.

Spring Boot надає широкий функціонал і серед його найкращих особливостей можна виділити такі, автоматична конфігурація та вбудовані контейнери сервлетів, як управління залежностями.

Щоб прискорити процес управління залежностями, Spring Boot неявно пакує необхідні сторонні залежності для кожного типу програми на основі Spring і надає їх розробнику за допомогою так званих starter-пакетів (spring-boot-starter-web, spring-boot-starter-data-jpa).

Starter-пакети – це набір дескрипторів залежностей включених у програму. Це надає універсальне рішення для пов'язаних зі Spring технологіям та позбавляє від лишнього пошуку прикладів налаштування і завантаження потрібних дескрипторів залежностей.

Наприклад, якщо ви хочете почати використовувати Spring Data JPA для доступу до бази даних, просто включіть у свій проект залежність spring-boot-starter-data-jpa і все буде готове (вам не доведеться шукати сумісні драйвери баз даних та бібліотеки Hibernate)

При створенні Spring web-додатока, потрібно додати залежності spring-boot-starter-web, яка підтягне в проект усі бібліотеки для розробки Spring MVC-додатків, таких як spring-webmvc, jackson-json, validation-api та Tomcat

Іншими словами, Spring Boot збирає всі загальні залежності та визначає їх в одному місці, що дозволяє розробникам просто використовувати їх, замість того, щоб винаходити колесо щоразу, коли вони створюють нову програму

Після вибору відповідного starter-пакета виконується автоматична конфігурація.

Якщо ви використовуєте spring-boot-starter-jdbc, Spring Boot автоматично реєструє біни DataSource, EntityManagerFactory, TransactionManager та зчитує інформацію для підключення до бази даних із файлу application.properties

Кожний Spring Boot web-додаток включає вбудований web-сервер. Розробникам тепер не треба турбуватися про налаштування контейнера сервлетів та розгортання програми на ньому. Тепер програма може запускатися сама, як виконуваний jar-файл з використанням вбудованого сервера.

3 ПРОЕКТУВАННЯ ТА РОЗРОБКА ВЕБ-ДОДАТКУ ТА ВИКОНУВАНОЇ СХЕМИ

Традиційно бізнес-процес (БП) визначається як логічно вибудувана послідовність дій по вирішенню конкретизованих завдань. З точки зору практики застосування БП наведемо таке визначення. Бізнес-процес - це набір заходів, які є необхідними і достатніми для того, щоб найбільше ефективно та оптимально зв'язати початок (проблему) і результат, що є вирішенням певної економічної, адміністративної або бізнес задачі.

Бізнес-процеси будуються за допомогою аналітичної диференціації загального процесу на елементарні складові (одновариантно описуються елементи процесу), з подальшою синтетичної інтеграцією (узагальненням) споріднених елементів процесу по функціональним, інтелектуального, технологічного, територіальною та іншими ознаками.

Таким чином, спочатку аналізується процес з точки зору виявлення його елементів, далі елементи класифікуються за заданими функціональними критеріями (кваліфікаційні вимоги, тривалість впливу, технічне забезпечення тощо), а потім будується система узагальнення елементів за критерієм ресурсовитрат для більш ефективного використання необхідного ресурсу. На завершення фіксується послідовність проведення операцій для даного БП з заданим (запитуваною, очікуваним) рівнем якості.

У розширеному варіанті опису БП також відображаються параметри необхідних технічних ресурсів, матеріальних ресурсів, параметр значущості (критичності) процесу (елемента) і т. д.

Через детальний опис бізнес-процесу ми виявляємо ті елементи й етапи, які можуть бути ключовими як з точки зору ефективності прийнятих рішень, так і з точки зору розподілу навантаження (у тому числі кваліфікаційної навантаження) між фахівцями.

Для створення ефективного продукту в страхуванні необхідно, щоб у ньому взяли участь фахівці:

- по залученню (аквизиционной діяльності, продажу, просування);

- облікові (бухгалтерія, податковий відділ, відділ актуарних розрахунків);

- андеррайтингу та розміщення ризику (перестрахування);

- врегулювання збитків.

Саме в процесі спільної роботи над продуктом фахівці різних підрозділів формують для себе більш об'єктивне уявлення про вимоги до продукту, які виставляють суміжні підрозділи, а також сукупності очікування, які клієнт пред'являє до якості продукту.

В рамках виконаної роботи було розроблено веб-додаток для автоматизації бізнес-процесів страхової компанії:

Далі на рисунках будуть представлені приклади бізнес-процесів у вигляді BPMN-нотацій та їх роз’яснень:
Перший процес, який слід взяти до автоматизації – це оформлення заявки про страховий випадок. Цей процес є одним із найрозповсюдженіжих у всіх страхових компаніях, тому доцільно автоматизувати його першим. Процес стартує і після його старту менеджер заповнює дані про заявку та клієнта на формі. Потім проводяться перевірки на рахунок того чи є даний клієнт в базі. Якщо такого клієнта не знайшлось то процес йде по альтернативному сценарію і відсилає результат в технічну підтримку та менеджеру клієнта. Якщо клієнта знайдено в базі, то потім заявка відправляється на погодження різних спеціалістів, а саме заповнюються висновки технічного спеціаліста, заповнюються висновки та оформлюється сума виплати згідно з умовами договору, та іншими факторами, а також потрібно підтвердження менеджера клієнта, у іншому випадку, менеджер за вимогою клієнта може відхилити походження та завершити розгляд заявки. Після успішного погодження, менеджер клієнта повідомляється про підтвердження заявки для повідомлення клієнта. Процес завершується успішно.

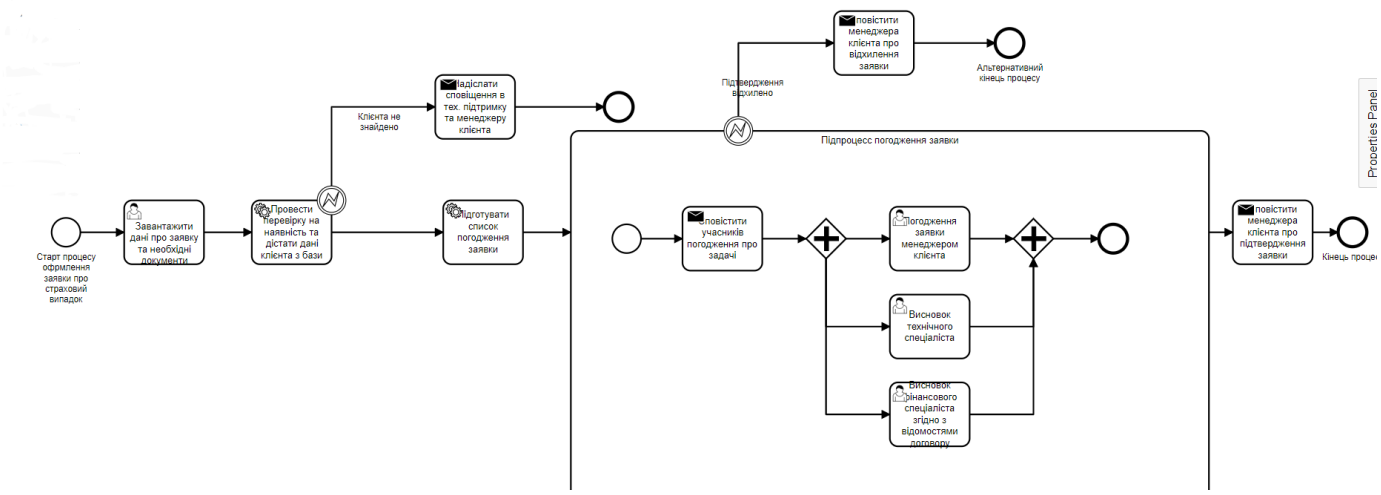

Рисунок 3.1 – Підтвердження заявки на оформлення страхового випадку.

Другий процес – це розрахунок ціни за пакет страхування, доволі розповсюджений бізнес-процес, так як перша відомість, яку захоче дізнатись клієнт – це ціна за пакет страхування. Після старту процесу менеджер клієнта вводить дані саме про клієнта. Потім за введеними даними автоматично розраховується попередня ціна. Після цього менеджеру приходить сповіщення з розрахованою попередньо ціною. Після цього менеджер повідомляє попередню ціну клієнту і може відмінити заявку і тоді процес завершиться за альтернативним шляхом. Або процес може піти далі, і на наступному етапі менеджер та спеціалісти розглядають страхову історію клієнта та інші параметри. Потім менеджера клієнта надсилають сповіщення про кінцеву ціну страхового пакету. Після цього клієнту доноситься кінцева ціна і він вирішує чи він згоден з ціною чи ні. Якщо так, то процес завершується успішно, в іншому випадку процес закінчується за альтернативним шляхом.

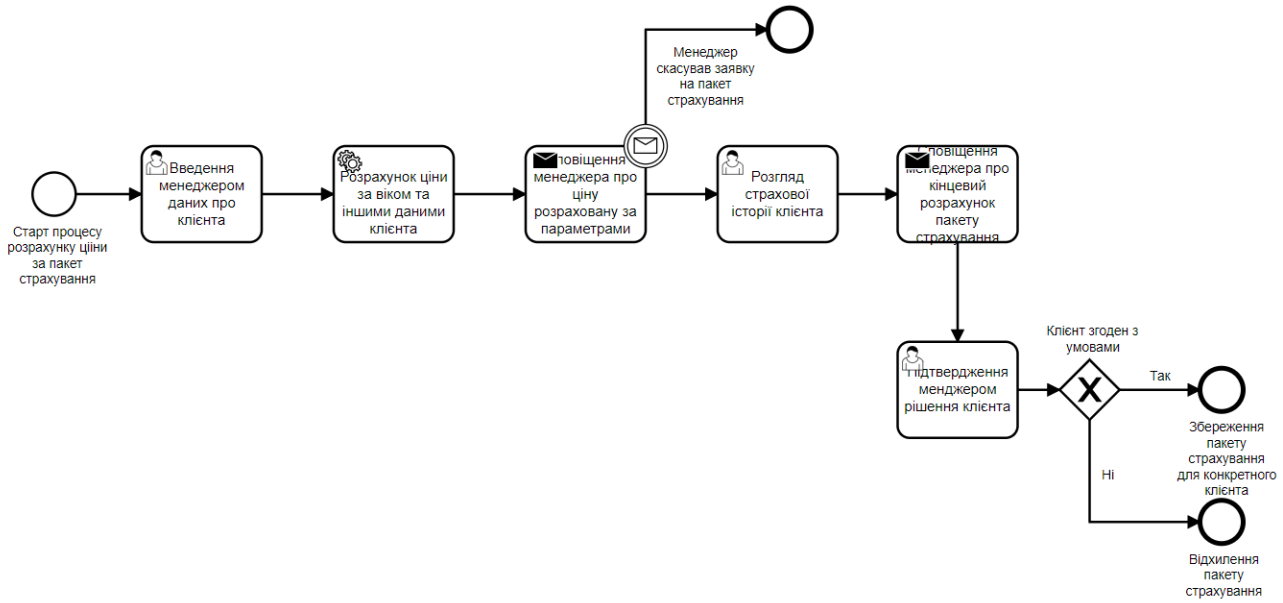

Рисунок 3.2 – Розрахунок ціни за пакет страхування.

Третій бізнес-процес зв’язаний з попереднім – це розрахунок та формування договору страхування. Процес починається з отримання повідомлення, тобто отримання заявки на розрахунок та формування договору, за згодою менеджера та клієнта або одразу після розрахунку ціни за пакет страхування або протягом декількох днів. Розраховуються умови договору, якщо потрібні додаткові умови, то менеджер додає їх до основних розрахованих умов. Також є альтернативний шлях, коли під час виконання процесу та визначення додаткових умов приходить сигнал про зміну обов’язкових умов, тоді процес відправляється до моменту перед розрахунком обов’язкових умов договору. Після того як основні та додаткові умови визначено, то на базі вимог, розраховується та формується договір. Після цього сформований договір надсилається менеджеру, після чого процес завершується.

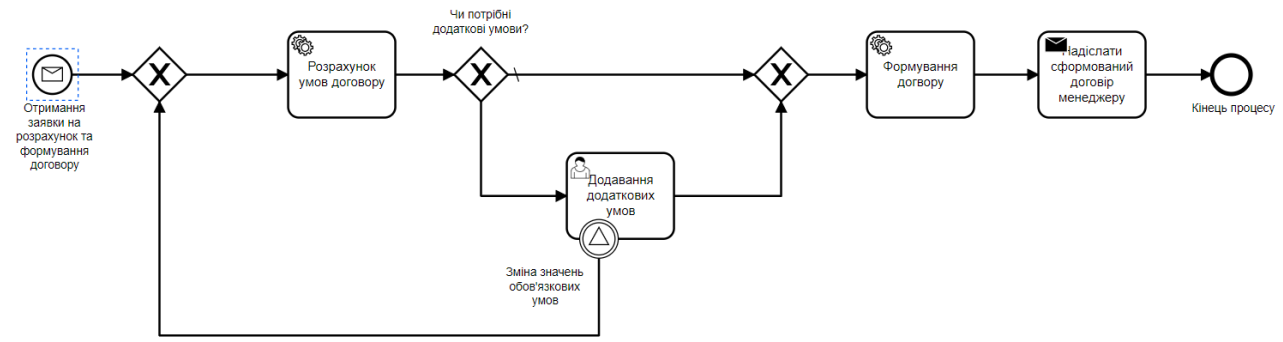

Рисунок 3.3 – Розрахунок та формування договору страхування.

Четвертий процес – теж доволі розповсюджений – це процес прийому на роботу нового співробітника. Після старту процесу співробітник відділ кадрів заповнює заявку даними наданими новим кандидатом, а також визначає чи потрібна йому додаткова співбесіда, якщо непотрібна, то процес іде далі, якщо навпаки – то призначається ще одна співбесіда, якщо кандидат не проходить додаткову співбесіду, то йому надсилається повідомлення про вдмову. Якщо кандидат пройшов технічну співбесіду, то процес повертається до основного «шляху» і далі формується список задач для різних відділів для прийому нового співробітника, а саме задачі для підтвердження та заповнення даних та документів бухгалтером, юристом та начальником відділу. Після цього співробітник зберігається в організаційній структрі і процес закінчується.

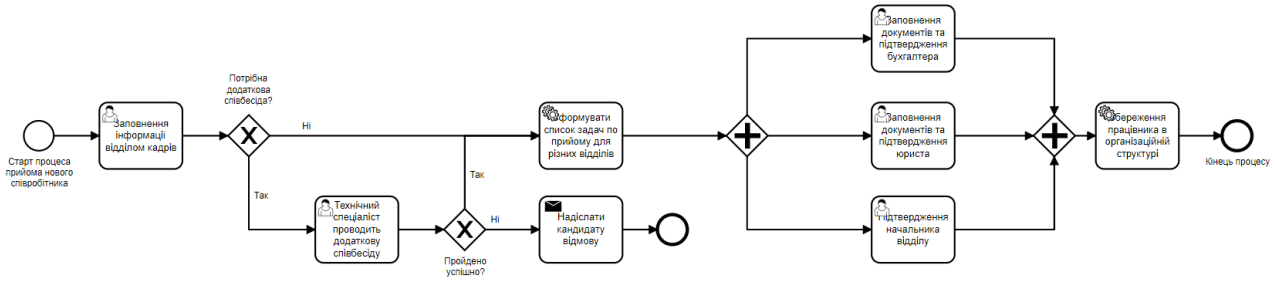

Рисунок 3.4 – Прийом на роботу нового співробітника.

І останній важливий процес – це проведення платежу по оформленому страховому випадку. Процес починається і одразу за допомогою сервісного завдання отримується інформація по страховому випадку. Далі бухгалтеру надсилається сповіщення про проведення платежу. Після цього платіж погоджується фінансовим відділом. Якщо клієнту за договором потрібно оплачувати франшизу, то дістається вартість франшизи з договору кілєнта, після чого платіж проводиться через зовнішні банківскі системи.

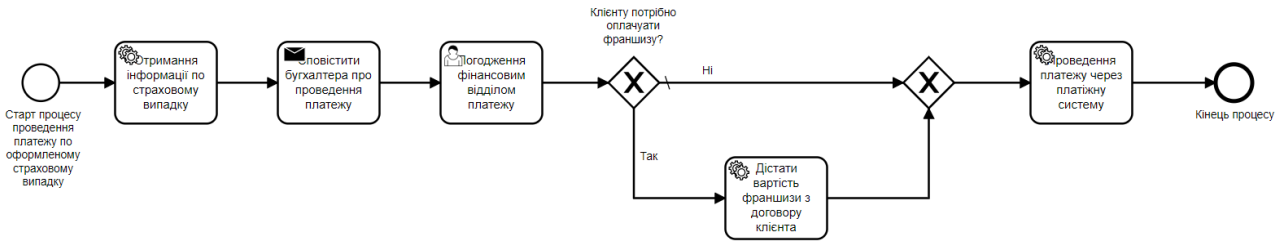

Рисунок 3.5 – Проведення платежу по оформленому страховому випадку

3.1 Опис бази даних додатку.

Далі необхідно для імітування організаційної структури треба створити реляційну базу даних, яка буде складатися з декількох таблиць. Основні з них – це користувач, пристрій та дія. Як СУБД буде використано PostgreSQL [10]. База даних буде мати наступну структуру, які наведені нижче у вигляді таблиць.

Таблиця 3.1 - Клієнт

Назва	Опис	Тип даних	Довжина	Первинний ключ	Зовнішній ключ
id	Унікальний ідентифікатор	INT	10	X	
email	Електронна пошта клієнта	VARCHAR	20		
middle_name	По-батькові клієнта	VARCHAR	30		
first_name	Ім’я клієнта	VARCHAR	25		
last_name	Прізвище клієнта	VARCHAR	25		
passport	Серія та номер паспорта клієнта	VARCHAR	8		
client_uuid	ІПН клієнта	VARCHAR	12		
birthday	Дата народження	DATE			

	клієнта				

Таблиця 3.2. Взаємодії з клієнтом (договори, страхові випадки)

Назва	Опис	Тип даних	Довжина	Первинний ключ	Зовнішній ключ
id	Унікальний ідентифікатор дії	INT	10	X	
action_type	Тип взаємодії	VARCHAR	20		
action_status	Статус взаємодії	VARCHAR	20		

Таблиця 3.3. Співробітник

Назва	Опис	Тип даних	Довжина	Первинний ключ	Зовнішній ключ
id	Унікальний ідентифікатор	INT	10	X	
email	Електронна пошта клієнта	VARCHAR	20		
middle_name	По-батькові клієнта	VARCHAR	30		

first_name	Ім’я клієнта	VARCHAR	25		
last_name	Прізвище клієнта	VARCHAR	25		
position	Посада співробітника	VARCHAR	25		
department	Відділ співробітника	VARCHAR	25		

Таблиця 3.4. Проміжна таблиця між взаємодіями та клієнтами

Назва	Опис	Тип даних	Довжина	Первинний ключ	Зовнішній ключ
id	Унікальний ідентифікатор дії користувача	INT	10	X	
action_id	Ідентифікатор дії	INT	10		X
user_id	Ідентифікатор користувача	INT	10		X

Таблиця 3.5 – Проміжна таблиця між співробітниками та клієнтами

Назва	Опис	Тип даних	Довжина	Первинний ключ	Зовнішній ключ
id	Унікальний ідентифікатор	INT	10	X	

	пристрою користувача				
device_id	Ідентифікатор пристрою	INT	10		X
user_id	Ідентифікатор користувача	INT	10		X

На рис. 3.10 зображена ER-діаграма спроєктованої реляційної бази даних, також її можна побачити у додатку.

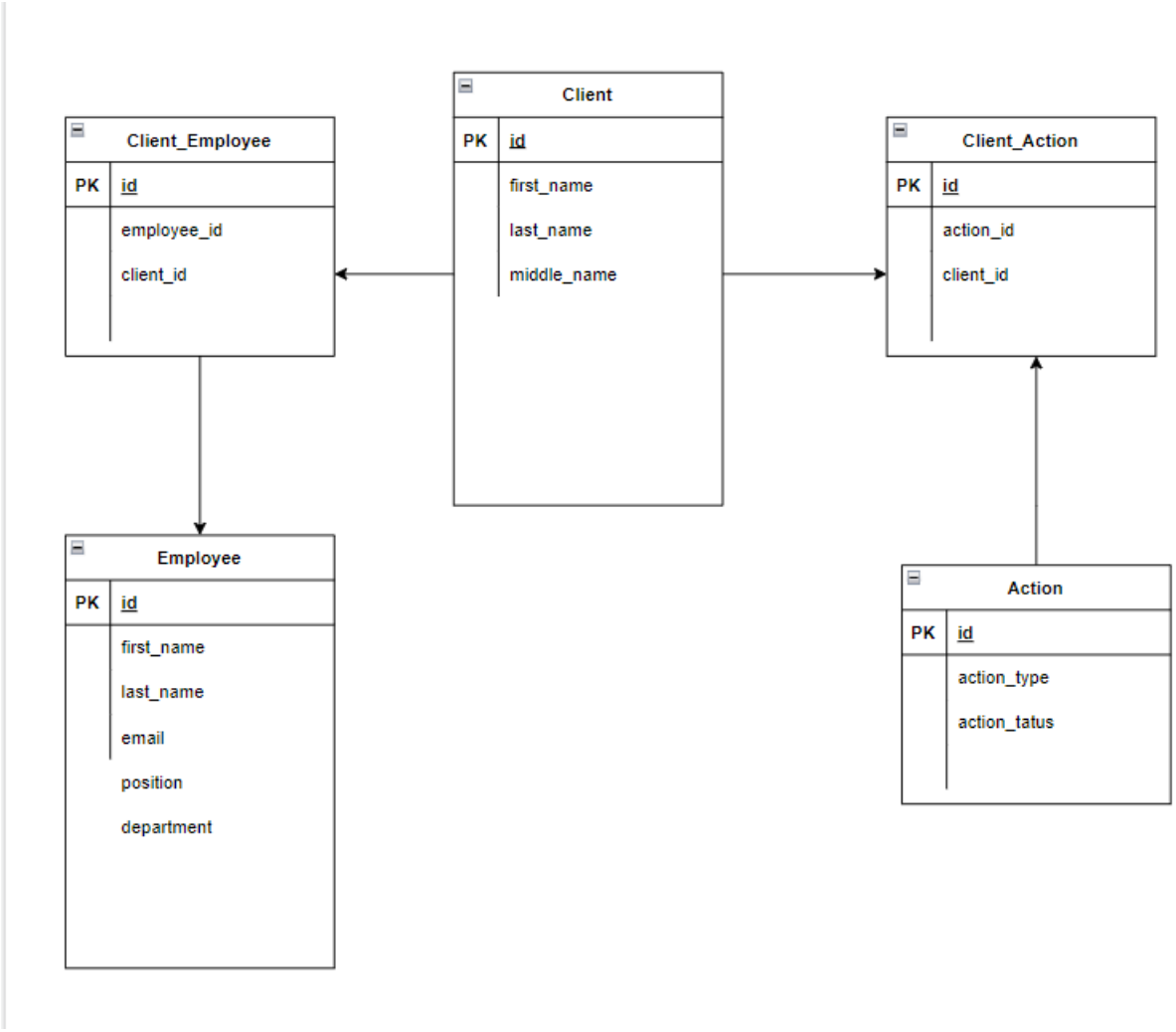

Рисунок 3.10 – ER-діаграма реляційної бази даних

Зв’язок між таблицями має вигляд багато до багатьох. Були додані таблиці, в яких буде зберігатись інформація про користувача, взаємодій, які можуть виконуватися з ним, та співробітників, які з ним взаємодіють. Також було додано

проміжні таблиці для реалізації зв’язку багато до багатьох між таблицями вказаними вище.

Далі розглянемо приклади інтерфейсу додатку для демонстрації функціоналу веб-сервіса за допомогою стандратних інструментів Camunda.

Для більш наглядного демонстрування далі буде надано знімки екрану з інтерфейсом веб-додатку.

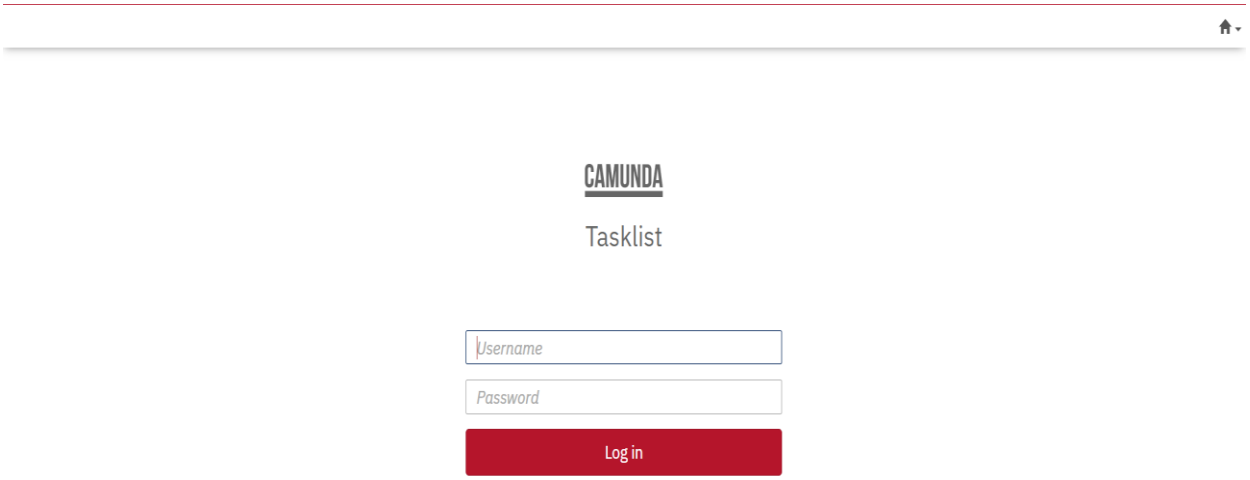

Рисунок 3.11 – Сторінка авторизації Camunda

Головна сторінка інструменту Camunda Task List слугує основним інтерфейсом для користувача. З неї можна стартувати процеси, дивитись список задач, як загальних, так і особистих, а також задачі, які доступні тільки для певної групи користувачів. Даний інтерфейс проілюстровано далі на рис. 3.12:

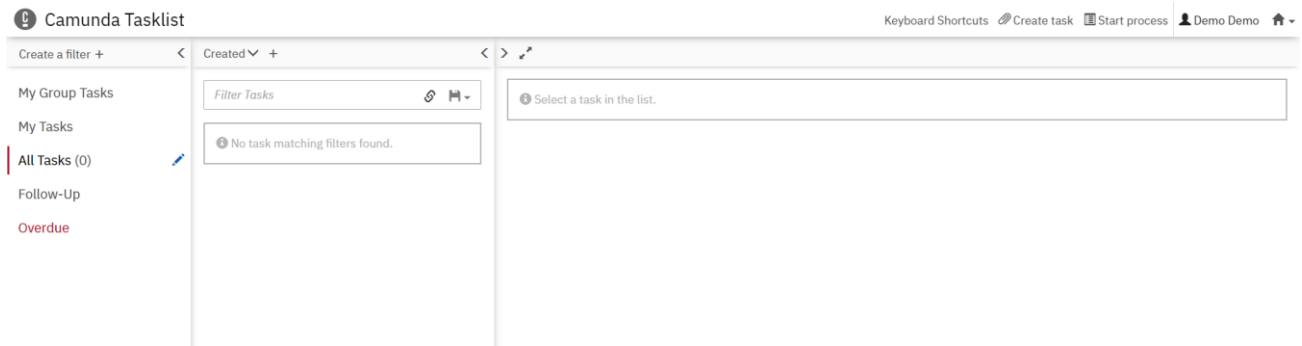

Рисунок 3.12 – Головна сторінка інструменту Camunda Task List

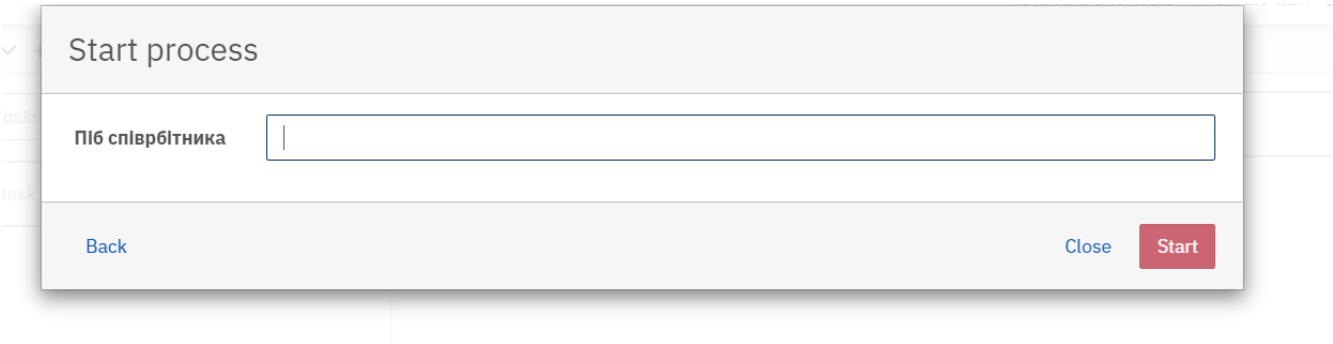

Рисунок 3.13 – Форма старту процесу по прийому нового співробітника

Після старту процесу в списку задач з'являється нова задача користувача, яку можна взяти в роботу. Також є можливість відфільтрувати список задач за обраними критеріями. Запис задачі в списку задач містить Назву задачі користувача, назву процесу, та дату створення задачі.

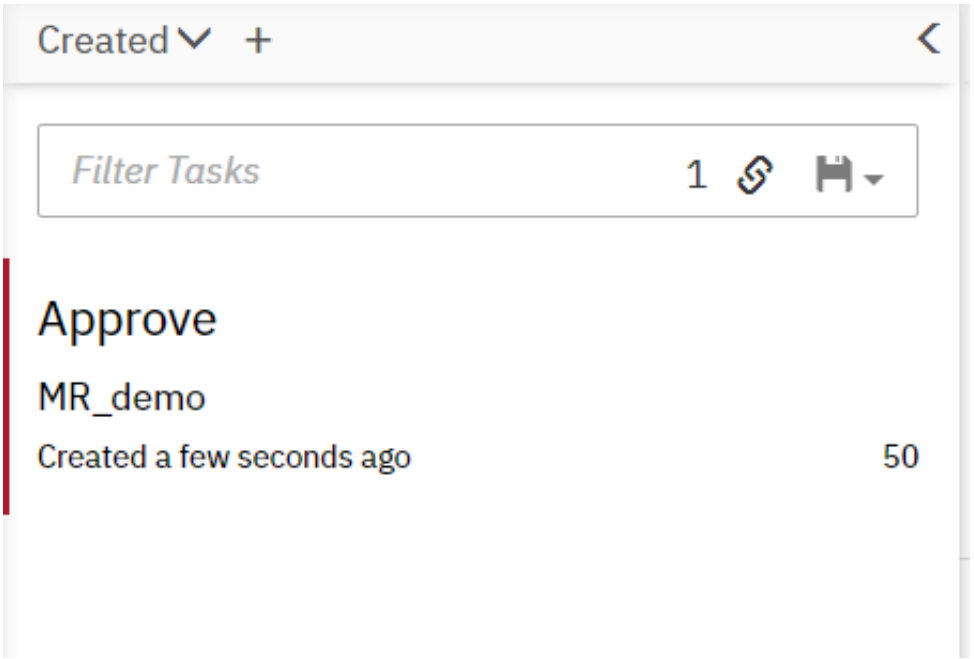

Рисунок 3.14 – Приклад списку задач

Далі після того як користувач переглянув список задач, в нього є можливість обрати і подивитись конкретну задачу. На рис. 3.15 наведено приклад форми користувача з процесу прийняття нового співробітника. На сторінці присутні назва задачі, назва процесу. Також є можливість встановити кінцеву дату виконання заявки, додати

групи користувачів, які можуть виконати дану заявку та прив’язати до заявку до авторизованого користувача, таким чином призначити користувача до виконання даного завдання користувача.

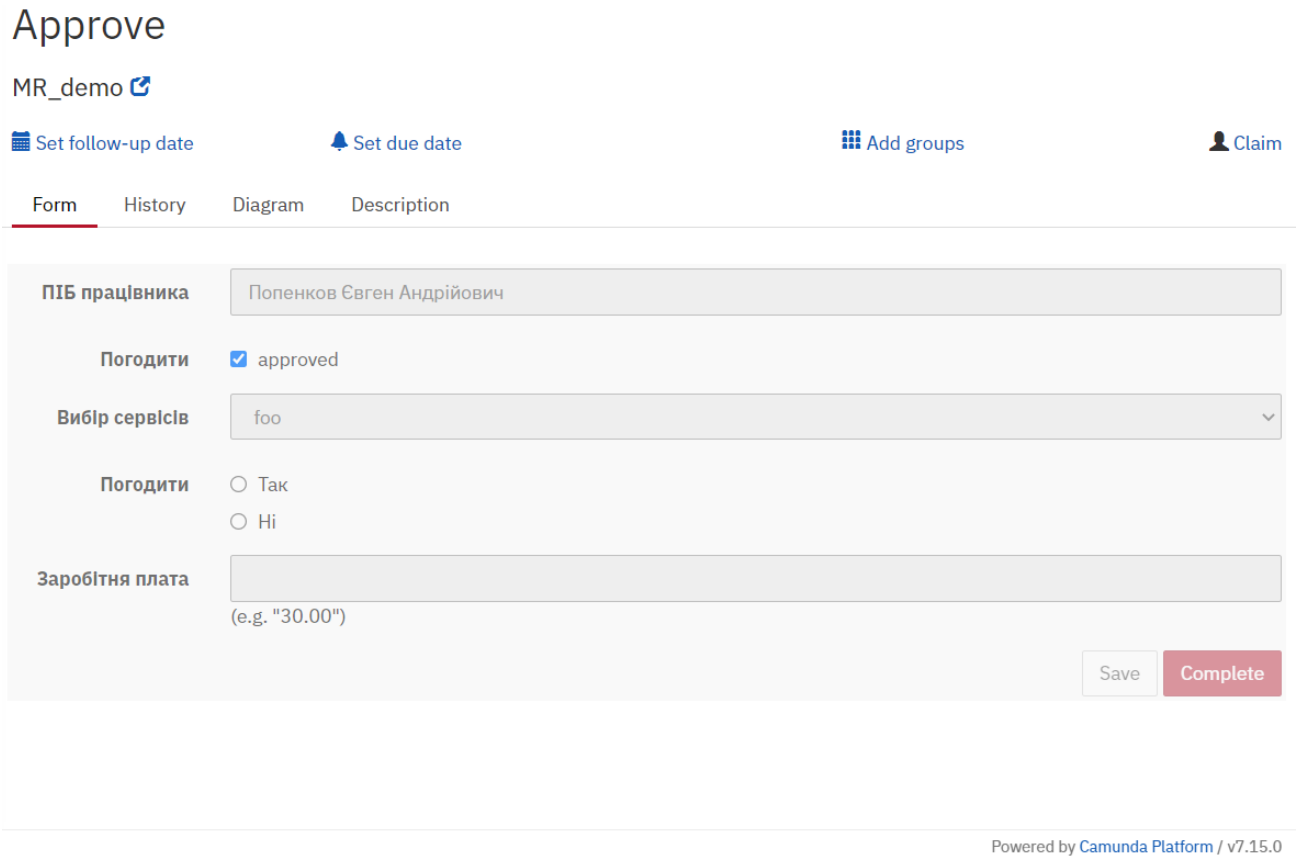

Рисунок 3.15 – Приклад сторінки для виконання задачі користувача

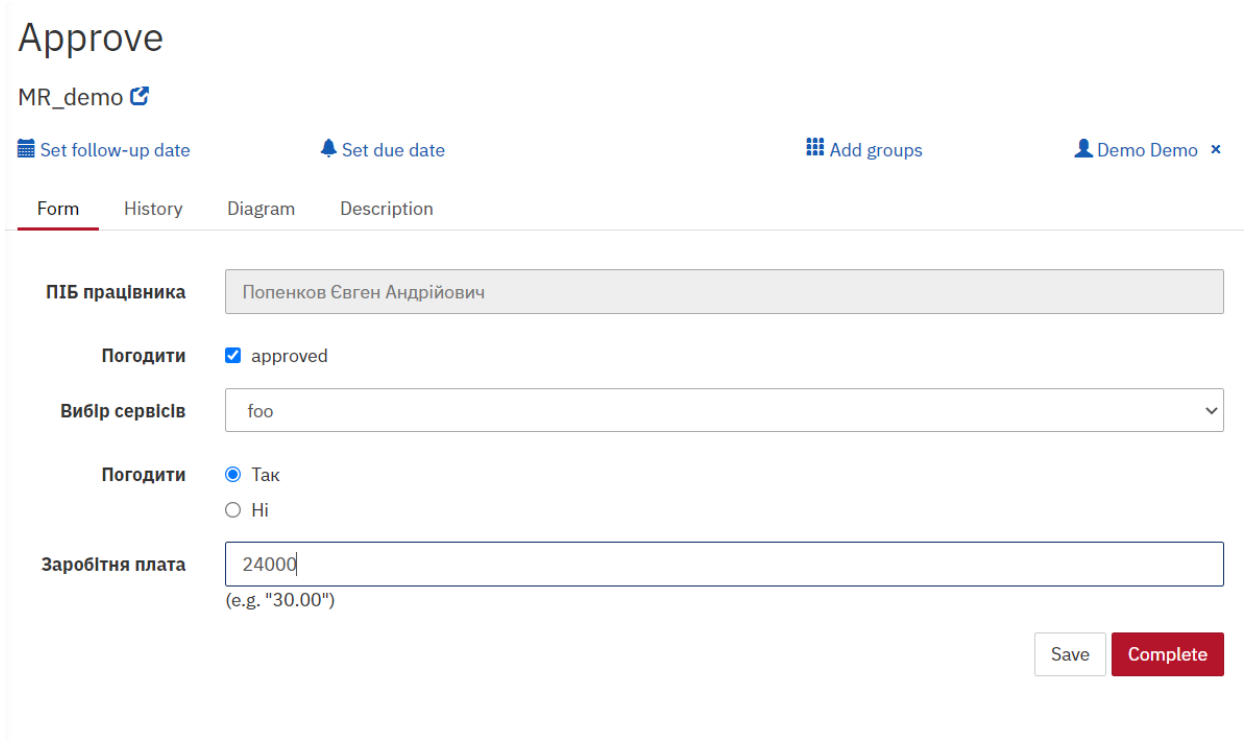

Рисунок 3.16 – Приклад заповненої сторінки для виконання задачі користувача

На рисунку 3.7 наведено сторінку з функціоналом завантаження документів для виконання задачі користувача, яка використовується в процесі прийняття нового співробітника, а також в процесі оформлення заявки про страховий випадок.

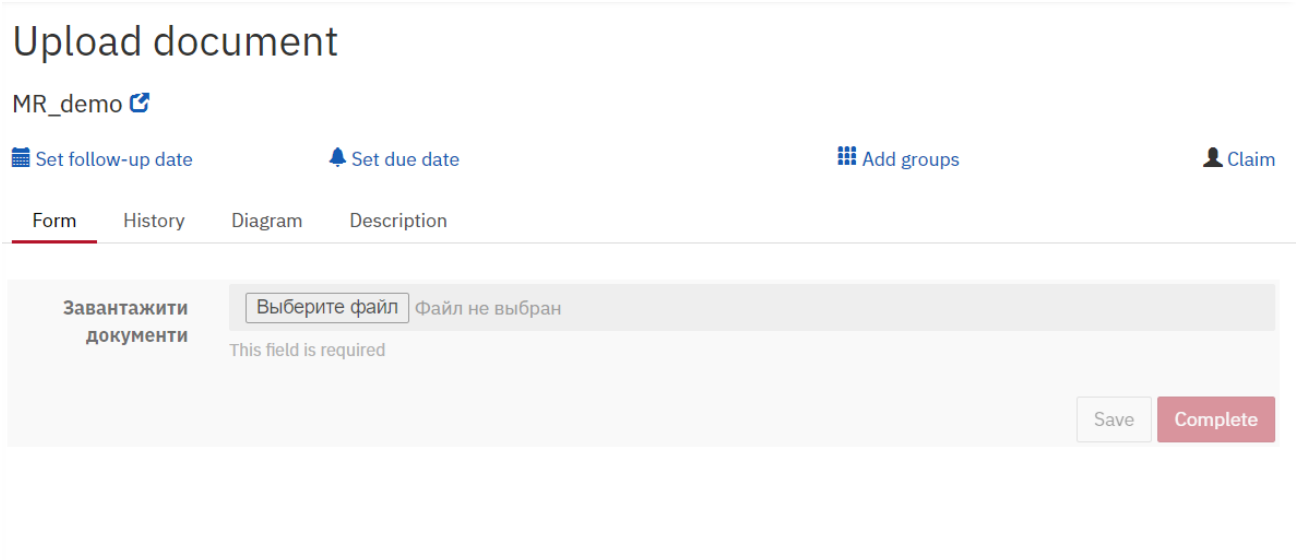

Рисунок 3.17 – Приклад сторінки з функціоналом завантаження документів для виконання задачі користувача процесу прийняття нового співробітника

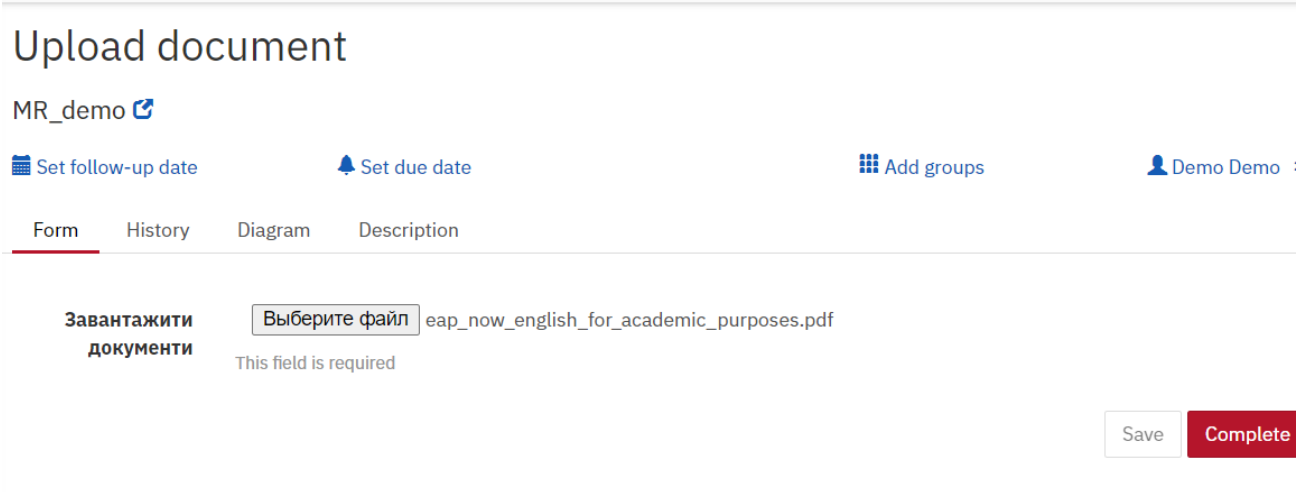

Рисунок 3.18 – Приклад заповненої сторінки з завантаженим документом для виконання задачі користувача

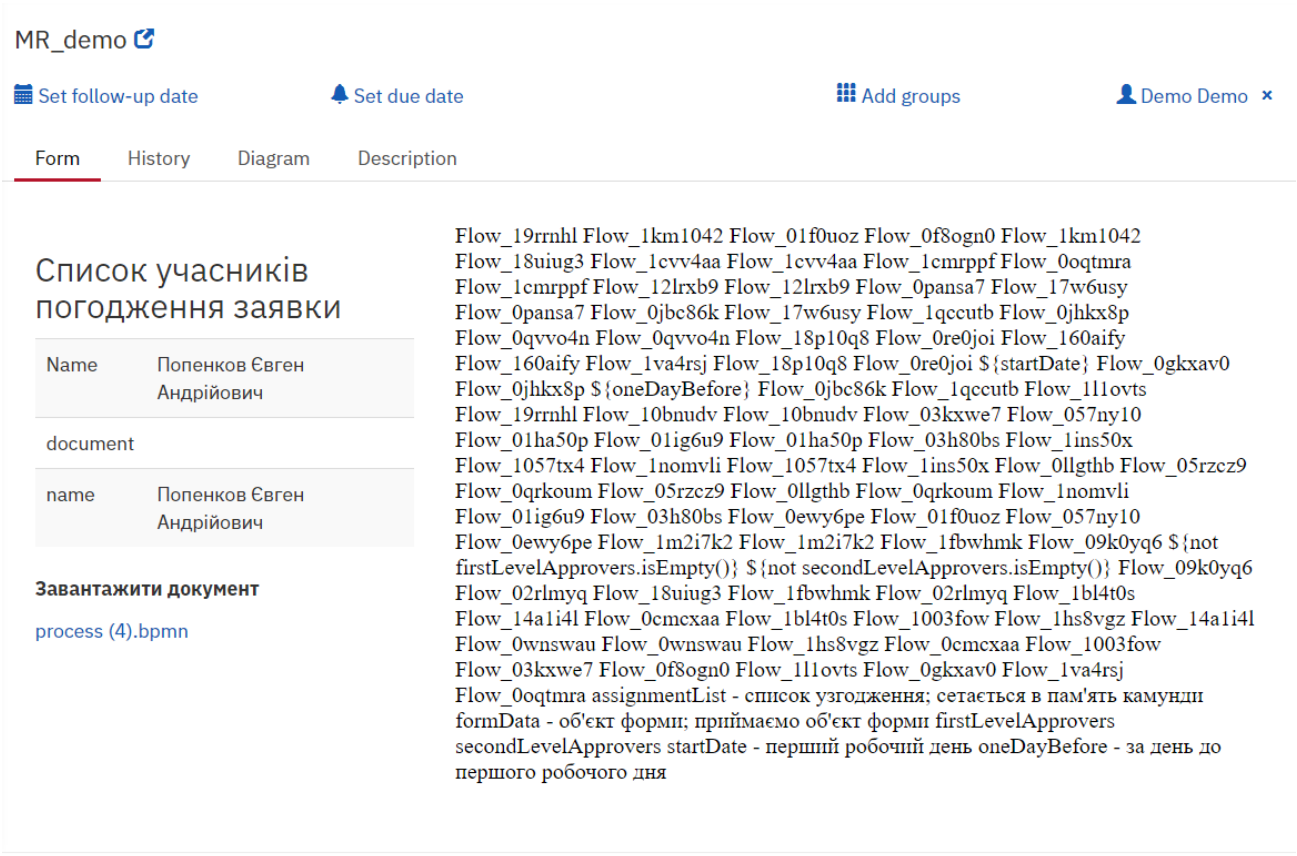

Рисунок 3.19 – Приклад заповненої сторінки погодження документу з завантаженим документом

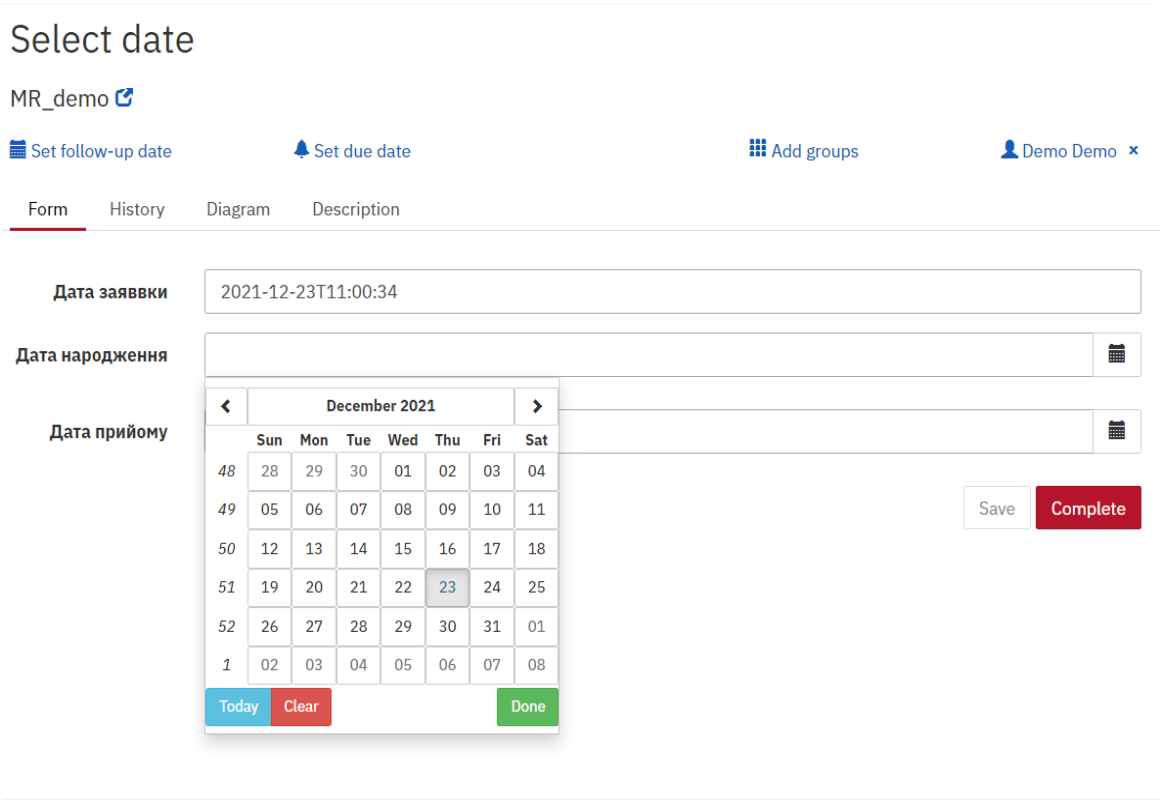

Рисунок 3.20 – Форма для заповнення менеджером відділу кадрів даних заявки по новому співробітнику

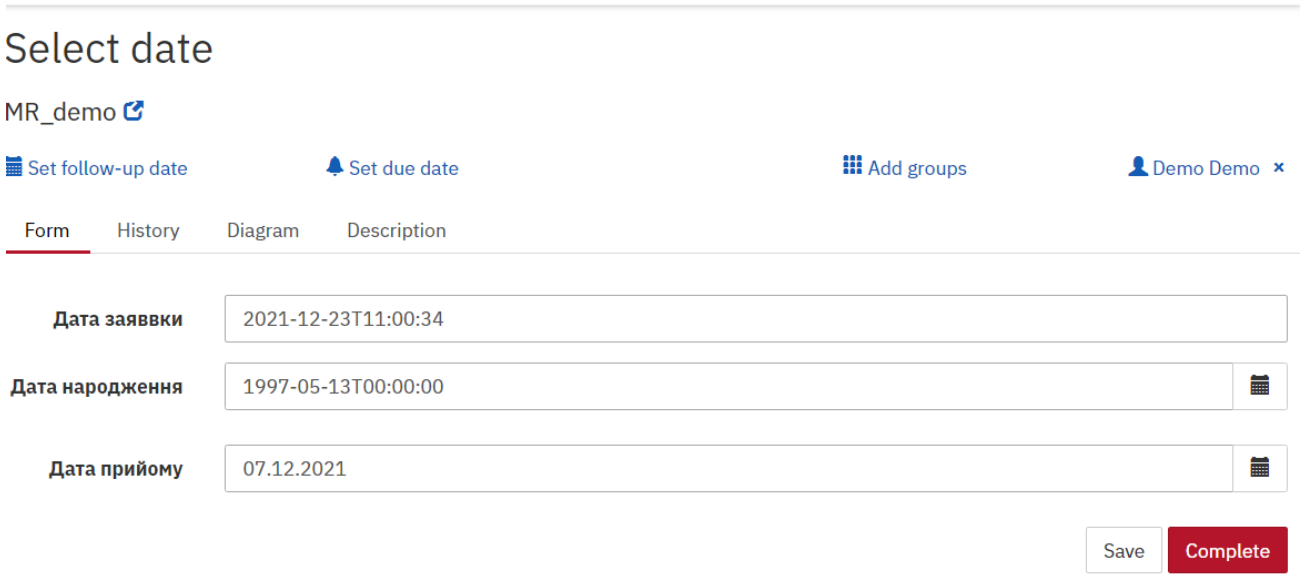

Рисунок 3.21 – Заповнена форма для форматування та редагування полів заявки для оформлення нового співробітника

Далі розглянемо декілька додаткових бізнес-процесів, які хоч і не є основними, проте мають теж важливе значення. Далі на рис. 3.22, 3.23 та 3.24 буде надано BPMN-схеми даних процесів.

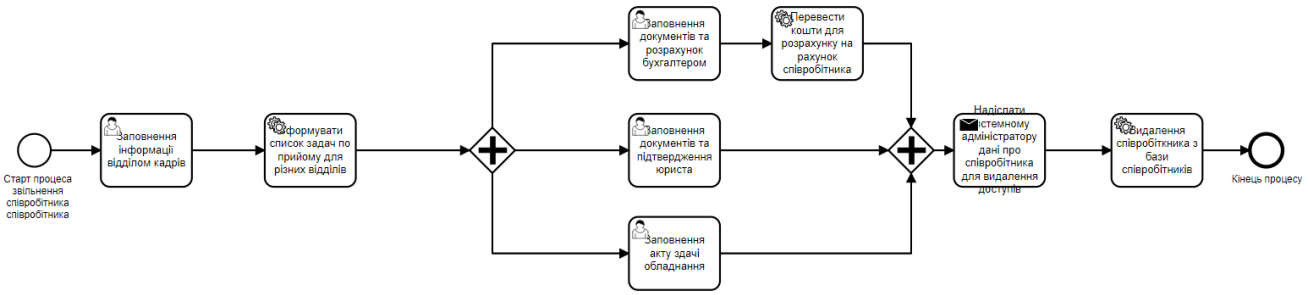

Рисунок 3.22 – Процес звільнення співробітника

Також після процесу розробки можна виділити наступну архітектурну схему, а саме схему роботи веб-додатку. Вона наглядно показує, що користувач взаємодіє з веб-інтерфейсом через інтерфейс додатку Camunda Task List, потім запит від користувача йде до веб-додатку, а далі в залежності що треба дістати: дані про користувачів, співробітників або відповідних дій, або ж потрібно дані про процес; -

то додаток направляє запит або в базу даних саме додатку, або в Camunda Process-Engine та до внутрішної бази даних процесів.

Висновки до розділу 3.

У даному розділі було описана реалізація інтерфейсу управління підсистемою безпеки розумного будинку на базі Web-додатку. Для реалізації системи використано мову програмування Java, Camunda.

Реалізація інтерфесу управління бізнес процесами страхової компанії базується на розділенні функціоналу системи на окремі незалежні один від одного модулі. Обрана архітектура додатку дозволяє зменшити ступінь залежностей між його компонентами, а також збільшити можливості для повторного використання реалізованого функціоналу.

Також у розділі було надано та розписано схеми бізнес різних бізнес процесів, які було автоматизовано за допомогою реалізації сервісних задач та Camunda-engine.

Графічний користувацький інтерфейс додатку виконаний у вигляді стандартних форм Camunda є доволі зручним та зрозумілим на інтуїтивному рівні.

Надано опис основних компонентів програмного забезпечення та наведено детальний опис роботи додатку. Створено інструкцію користувача програмного забезпечення, що показує послідовність дій при роботі з додатком.

Більше того описано та реалізовано додатковий процес по звільненню співробітника з страхової компанії.

Також описано та створено базу даних, вказано перелік таблиць, які використовуються в додатку, а також надано зменшену копію ERD-моделі бази даних системи. Саму ERD-діаграму для подальшого ознайомлення можна знайти в додатках.

4 РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

4.1 Опис ідеї проекту

Ідея проекту даної роботи - розробити систему, яка автоматизує бізнес-процеси, що призведе до економії часу при виконанні бізнес-процесів, це також призведе до зменшення витрат, та дозволить полегшити менеджерам роботу з клієнтами і не тільки. Це є надзвичайно позитивним фактором так як у сфері інформаційних технологій більша частина бюджету витрачається на робітників та на години їх роботи.

Користувачі системи мають можливість створювати документи та виконувати операції з ними.

Узагальнення цих ідей можна побачити в таблиці 4.1.

Таблиця 4.1

Вигоди для користувача
Скорочення часу, що використовується на комунікацію робітників
Можливість детального проектування у неформальному вигляді

Основні вимоги до стартап-проектів, як правило, полягають у наступному:

- аналіз ринку на наявність аналогічних рішень;
- ознайомлення можливих споживачів та партнерів зі стартап-проектом;
- обрати найкращий час для подання інформації про стартап-проект;
- забезпечити процеси обробки інформації при реалізації проекту;
- зберегти результати обробки інформації в необхідному вигляді
-

Загальні обмеження на роботу з інформацією можуть бути представлені якгрупа

характеристик, які дійсно належать сторонам стартап-проекту, а саме:

- об'єкту (джерелу інформації);
- медіатору (засобу отримання, обробки, виробництва і передачі інформації, посереднику між суб'єктом і об'єктом);
- суб'єкту (активній стороні діяльності, одержувачу і перетворювачу інформації).

Складність отримання інформації на сьогодні обумовлена її розподіленням та фрагментарністю, а також різнопрофільним заповненням.

Зараз на ринку існує декілька основних сервісів для колаборативного менеджменту контенту, доведеться конкурувати і з ними, а не тільки з тими, що пропонують рекомендації. Тому із найкрупніших конкурентів можна виокремити Symbol, RM-SKLAD.

4.2 Аналіз потенційних техніко-економічних переваг

Аналіз сильних, слабких та нейтральних сторін стартап-проекту визначено в таблиці 4.2. Перевагами системи є автоматизація найбільш розповсюджених бізнес-процесів.

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

Техніко-економічні характеристики ідеї	товари/концепції конкурентів			Слабка сторона	Нейтральна сторона	Сильн а сторона
	Мій проект	IBM BPM	Pega			

Техніко-економічні характеристики ідеї	товари/концепції конкурентів			Слабка сторона	Нейтральна сторона	Сильна сторона
Можливість обробляти одночасно велику кількість процесів	Так	З значними уповільненням системи	З значними уповільненням системи			+
Вартість експлуатації	Невелика	Велика	Невелика			+
Кількість компонентів	Велика	Велика	Середня		+	
Надаються додатки для користування та адміністрування	Є	Є	Немає		+	

Технологічний аудит ідеї проекту

Список технологій реалізації програмного комплексу наведено в таблиці 10.3.

Таблиця 4.3 - Технологічна здійсненність ідеї проекту

Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій

Обрана технологія реалізації ідеї проекту:			
	Camunda, Spring Boot	Технології наявні, їх потрібно обєднати міжсобою	Технології наявні, їх потрібно обєднати міжсобою
Для маштабованого рівня для бізнес логіки Camunda, Spring Boot			

Аналіз ринкових можливостей запуску стартап-проекту

Таблиця 10.4 - Попередня характеристика потенційного ринку проекту

Показники стану ринку (найменування)	Характеристика
Кількість головних гравців, од	5
Загальний обсяг продаж, грн/ум.од	~~22~~
Динаміка ринку (якісна оцінка)	Зростає
Наявність обмежень для входу (вказатихарактер обмежень)	Недискримінаційніякісні

Визначення потенційних груп клієнтів

Таблиця 4.5 - Характеристика потенційних клієнтів стартап-проекту

Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів дотовару
Бажання зменшення часу роботи програміста та приріст швидкості реалізації додатків	1. Малий середній та великий бізнес галузі IT	1. Ведення бізнесу на платформах інтернет-ресурсів	Можливість роботи з великою кількістю даних

Аналіз ринкового середовища

У таблиці 4.6 знаходять фактори, яка містить загрози для компанії,заводу, фактори можливостей – висока зацікавленість у таблиці 4.6

Таблиця 4.6 - Фактори загроз

Фактор	Зміст загрози	Можлива реакціякомпанії

Крадіжка інтелектуальної власності	Крадіжка ідеї або ключової інтелектуальної інновації	Відсудження прав інтелектуальної власності Забезпечення якісного захисту інформації Зміна методики групування даних
Відмова компанійу співпраці	Керівництво компанії не погоджується на співпрацю	Пропозиція більш вигідних умов співпраці

Таблиця 4.7 - Фактори можливостей

Фактор	Зміст можливості	Можлива реакціякомпанії
Висока зацікавленість користувачів	Обіг використання становить більше 1000 запитів в день	Підтримка стабільноїроботи системи та проведення масштабування системи

Фактор	Зміст можливості	Можлива реакція компанії
Успішна маркетингова політика	В результаті проведеної маркетингової	Підтримка стабільної роботи системи та проведення

	політики отримана висока зацікавленість користувачів	масштабування системи Збільшення цін на використання сервісу Використання подібної маркетингової стратегії надалі для залучення нових користувачів
Ліквідація конкурента	Конкурент ліквідував свою компанію у результаті власного бажання або зовнішніх чинників	Проведення маркетингової кампанії для монополізації ринку

4.3 Аналіз пропозиції

Ступеневий аналіз конкуренції описано в таблиці 4.8

Таблиця 4.8 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляєтьсядана характеристика	Особливості конкурентного середовища В чому проявляється дана характеристика Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможно

		ю)
Тип конкуренції -чиста	Немає чітко відокремленого лідера серед конкурентів, та відповідно мала їх кількість	Сприятливо впливаєна розвиток проекту
Рівень конкурентної боротьби - міжнародний	Сфера зачіпає практично всі країни світу	Можливий швидкий вихід на світові ринки
За галузевою ознакою - внутрішньогалузева	Загроза появи нових конкурентів Ринкова влада споживачів Висока потреба у товарі	Інформування ринку щодо якості використовуваної новаторської технології Пропозиція гнучких цін
За видами товарів - товарно-родові	Надання різних сервісіводного виду	Маркетингова політика
Цінова	Використання цін для покращення	Зменшення вартості сервісу каналів розподілу

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари замінники
	Прямих конкурентів немає, так як інтеграції на бази камунди доволі новий тип рішень і в Україні майже немає офіційно сертифікованих компаній по наданню даних послуг	Цінністьідеї та її якісна реалізація	Немає	Споживачі прямо впливають на успішність продукту, оскільки сплачуються я кошти за розробку інтеграцію та встановлення	Конкуренти можуть стати дешевшими або стануть надавати якісніші послуги
Висновки	відсутність монополії, проте конкурент на боротьбазначна за рахунок хорошої якості проектів	Є можливість виходу на ринок	Немає постачальників	Так, реалізація потреб клієнтів вирішує успішність проекту	Достатньо важко предентувати на клієнтів, які вже використовують інший готовий продукт

Перелік факторів конкурентоспроможності

Навіть при врахуванні конкурентів розроблена система має низку переваг,які наведені в таблиці 4.10.

Таблиця 4.10 — Обґрунтування факторів конкурентоспроможності

Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівнянняконкурентних проектів значущим)
У конкурентів наявні функціональні проблеми реалізації	Часткова або повна неможливістьпрацювати з великою кількістю даних
Ідея	розвиток ідеї колаборативної роботи
Швидкий вихід на ринок	Достатньо випустити мінімальну базову працездатну версію продукту на ринок дляйого використання
Цінова політика	Отримання прибутку здійснюється за рахунок гнучкої моделі оплати

Аналіз сильних та слабких сторін стартап-проекту

За наведеними факторами конкурентоспроможності в таблиці 10.10 було виконано порівняльний аналіз сильних та слабких сторін програмного рішення (стартап-проекту), який відображено в таблиці 4.11.

Таблиця 4.11 — Порівняльний аналіз сильних та слабких сторін системи

Фактор конкурентоспроможності	Бали1-20	Рейтинг товарів-конкурентів у порівнянні із системою управління процесами					
		-3	-2	-1	0	+1	+2
У конкурентів наявні функціональні проблеми реалізації	19		+				
Ідея	12				+		
Швидкий вихід на ринок	16			+			
Цінова політика	15			+			

SWOT-аналіз

Складений SWOT-аналіз на основі вище отриманих даних відображенов таблиці 4.12.

Таблиця 4.12 — SWOT-аналіз стартап-проекту

Сильні сторони:	Слабкі сторони:
– можливість роботи з великою кількістю інформації – розвинута система структурування – широкий спектр функціоналу; – невелика кількість сильних конкурентів;	– потреба в рекламі

Опис цільових груп потенційних клієнтів

В таблиці 4.14 наведено характеристики потенційних клієнтів.

Таблиця 4.14 - Характеристика потенційних клієнтів стартап проекту

Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
IT команди	Висока	85%	Незначна	Просто
Власне використання	Висока	75 %	Незначна	Просто

Формування базових стратегій розвитку

В таблиці 4.15 визначено базову стратегію розвитку

Таблиця 4.15 — Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конку-рентоспроможні позиції	Базова стратегія розвитку
Пропонування програмного комплексу для IT команд	Диференційований маркетинг	Здатність пропонувати унікальний функціонал	Стратегія диференціації

Базовою стратегією для стартап-проекту є стратегія диференціації, що означає надання програмному комплексу властивостей, які будуть відрізняютьйого від конкурентів.

Вибір стратегії конкурентної поведінки

В таблиці 4.16 наведено базові стратегії конкурентної поведінки.

Таблиця 4.16 — Визначення базової стратегії конкурентної поведінки

Чи є проект першопрохідцем наринку?	Чи буде компанія шукатинових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
Ні	Залучати нових та забирати існуючих	Віджети та графічне оформлення	Стратегія велику лідера

На основі потреб споживачів з обраних сегментів, а також в залежності від обраної базової стратегії розвитку в таблиці 4.17 наведено визначені стратегії позиціонування.

Таблиця 4.17 — Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну
Швидкодія та зручність	Стратегія диференціації	Розширення базового функціоналу	Якісний продукт, зрозумілий у використанні

Розробка маркетингової програми стартап-проекту

Першим кроком до формування маркетингової концепції товару, якийотримує споживач. Для цього у таблиці 4.18 підсумовано результати попереднього аналізу конкурентоспроможності товару.

Формування маркетингової концепції

В таблиці 4.18 визначено ключові переваги концепції потенційного товару

Таблиця 4.18 – Визначення ключових переваг концепції потенційноготовару

Потреба	Вигода, яку пропонуєтовар	Ключові переваги перед конкурентами (існуючі аботакі, що потрібно створити
Швидкодія	Стабільна швидкість завантаження сторінки та плавна її робота	У існуючих додатків існує проблема значного уповільнення додатку при великій кількості компонентів
Потрібнапам'ять	Оптимізація займаного простору	У існуючі додатки потребують відносто великих об'ємів пам'яті для роботи з великою кількістю елементів

Наступним кроком є визначення цінових меж представлених в таблиці 10.15,якими необхідно керуватись при встановленні ціни на товар

Висновки до розділу 4

Під час проведення дослідження магістерської дисертації у якості стартап-проекту було створено відповідну аргументовану документацію. Протягом усього процесу дослідження було досягнуто повне розуміння цілей проекту, його переваги та недоліки, які в свою чергу дали можливість точніше визначити мету проекту та користувацьку аудиторію програмного застосунку, визначити можливі методи монетизації, маркетингову стратегію для IT команд та власноговикористання.

На даний час попит на такий програмний комплекс може бути не дужевисоким через специфіку області застосування.

Перевагами проекту є:

- доволі просто при розробці;
- просто у використанні;
- кросплатформність;
- зручний інтерфейс користувача; оптимізованість додатку при великому навантаженні

ЗАГАЛЬНІ ВИСНОВКИ

При виконані даного індивідуального завдання в межах переддпипломної практики було проаналізовано процес автоматизації управління бізнес-процесами страхової компанії та визначено основні проблеми, які на сьогодні існують при реалізації даного процесу. Також було переглянуто і описано декілька варіантів існуючих реалізацій BPMN-інструментів та описано основні їх призначення. В результаті було обрано пакет технології Camunda-bpmn, на базі якого буде проведено автоматизацію бізнес-процесів. Було описано систему-прототип на основі якої запропоновано реалізацію. Було розроблено функціональні вимоги до розроблювальної системи. І для більшої зручності та наочності було складено декілька різних схем, які описують певні бізнес-процеси, які автоматизуються.

Потім було реалізовано веб-додаток для автоматизації бізнес-процесів страхової компанії. Додаток працює коректно. За допомогою утиліти Camunda Optimize було перевірено навантаженість на різні процеси. Найбільш навантаженим виявився процес оформлення заявки на страховий випадок.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Process Modeling Notations and Workflow Patterns Архівовано 6 липень 2010 у Wayback Machine., paper by Stephen A. White of IBM Corporation (2006)

2. [Електронний ресурс]. URL: https://habr.com/ru/post/244261/

3. Карлова Н.О. Реінжиніринг системи продажу страхових продуктів. Київський національний економічний університет імені Вадима Гетьмана, 2008.

4. Silver, Bruce (2011). BPMN Method and Style, 2nd Edition. Cody-Cassidy Press. ISBN 0982368119.

5. [Електронний ресурс]. URL: https://docs.camunda.org/manual/7.16/

6. [Електронний ресурс]. URL: https://tproger.ru/articles/modelirovanie-biznes-processov-praktika-ispolzovanija-camunda-bpm-v-java-razrabotke/