УДК 004.428:65.011.56

В.В. Герасимов, А.Я. Беличенко

ЦЕЛЕСООБРАЗНОСТЬ ИСПОЛЬЗОВАНИЯ БИЗНЕС-ПРОЦЕССОВ В КОРПОРАТИВНЫХ ПРИЛОЖЕНИЯХ

Аннотация. Проведено описание особенностей и взаимодействия с Java-кодом и сервисами технологии Apache Activiti, которая наряду с поддержкой BPMN2 (Business Process Model and Notation — система условных обозначений для моделирования бизнес-процессов) предлагает возможности настраивания и распределения транзакций, сохранения истории прохождения процессов. Использование таких систем дает такие преимущества как визуальное моделирование, поддержка версионности и гибкости конфигурирования бизнес-процессов. Приведены примеры использования рассматриваемой технологии.
Ключевые слова: Java, Spring, Activiti, BPMN.

Постановка проблемы. Одной из причин популярности языка Java есть возможность писать корпоративные приложения, благодаря чему очень многие крупные предприятия используют именно Java EE. Однако прогресс не стоит на месте и в связи с плотным переплетением интернета с нашей жизнью платформа Java EE стала несколько терять актуальность. Главным ее конкурентом стал фреймворк Spring [1], который предоставляет хорошо документированные и легкие в использовании средства решения проблем, возникающих при создании приложений корпоративного масштаба. Однако без использования бизнес-процессов построение максимально оптимизированного и гибкого приложения невозможно. Согласно “Google Trends” [2], количество запросов “BPM” постоянно увеличивается. Тем не менее, сходу сказать, как использование бизнес-процессов отразится на Вашем коде, не выйдет. Попробуем разобраться в этом вопросе, используя самую популярную нотацию BPMN, которая лежит в основе фреймворка Activiti [3].

Целью данной работы является оценивание пользы использования бизнес-процессов в современых корпоративных приложениях, рассмотрение фреймворка Activiti, его возможностей и синхронизации с Java.

Основная часть. BPM (business process management) — класс про-

 ISSN 1562-9945

граммных продуктов, которые помогают управлять бизнес-процессами организации. Использование таких систем дает такие преимущества, как визуальное моделирование, поддержка версионности и гибкость конфигурирования бизнес-процессов, а также взаимодействие с пользователем для выполнения ручных работ. В 2005 году путем слияния двух компаний Business Process Management Initiative (BPMI) и Object Management Group была разработана система условных обозначений для моделирования бизнес-процессов — BPMN (Business Process Model and Notation). BPMN уверенно захватила лидерство среди аналогов и спустя всего лишь четыре года после создания использовалась примерно в половине случаев, а после выхода в 2011 году версии BPMN 2.0 и вовсе стала самой популярной нотацией, а показатель её использования и вовсе перевалил за отметку в 60 %.

Именно основываясь на спецификации BPMN 2.0, компания Alfresco под лицензией Apache разрабатывает бесплатный фреймворк Activiti, в создание которого вложили свой вклад Том Байенс и Джорам Баррез — авторы конкурирующего проекта jBPM, которых удалось переманить.

Главными достоинствами Apache Activiti, помимо поддержки BPMN2, являются такие возможности, как настраивание и распределение транзакций, сохранение истории прохождения процессов. Кроме того, история сохраняется вместе с временными засечками, что позволяет следить за ходом выполнения процесса и по возможности оптимизировать его.

Сами бизнес-процессы пишутся с помощью XML-разметки, что является несомненным плюсом, поскольку не требует изучения нового языка. Кроме того, для лучшего восприятия можно использовать бесплатный плагин для IDE Eclipse, который является визуальным редактором.

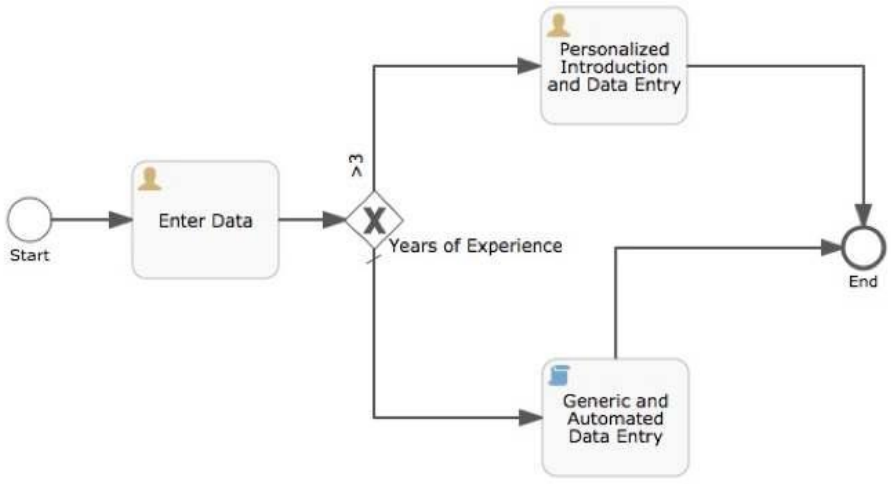

Рисунок 1 - Внешний вид бизнес-процесса

Activiti можно использовать как самостоятельное приложение, так и как библиотеку для обработки бизнес-процессов. В данной работе будет рассматриваться интеграция Activiti и Spring [4].

Для подключения Activiti в уже созданный проект, при условии, что Вы используете популярный фреймворк для автоматизации сборки проекта — Apache Maven, достаточно в файл зависимостей pom.xml добавить следующие зависимости [5]:

```
<dependency>
    <groupId>org.activiti</groupId>
    <artifactId>activiti-engine</artifactId>
    <version>6.0.0</version>
</dependency>
<dependency>
    <groupId>org.activiti</groupId>
    <artifactId>activiti-spring</artifactId>
    <version>6.0.0</version>
</dependency>
```

Эти две зависимости обязательны, так как они добавляют ядро Activiti и утилиты для интеграции со Spring'ом, а следующая подключает базу данных H2, и необязательна, если вы используете в своем проекте другую базу данных, которую, тем не менее, предварительно подключили в pom.xml:

```
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <version>1.4.193</version>
</dependency>
```

Подключение базы данных — это такой же важный момент, как и подключение всего Activiti, так как без хранения всей информации фреймворк теряет свою силу. Следующий шаг — добавление в файл конфигураций технологию Activiti. Компонент `ProcessEngineConfiguration` добавляется следующим образом:

```
<bean id="processEngineConfiguration" class="org.activi-
ti.spring.SpringProcessEngineConfiguration">
```

Этот компонент используется для построения `ProcessEngine` — центральной точки доступа ко всем функциональным возможностям Activiti. Добавление самого `ProcessEngine`:

```
<bean id="processEngine"
class="org.activiti.spring.ProcessEngineFactoryBean">
```

```
    <property name="processEngineConfiguration"
ref="processEngineConfiguration" />
    </bean>
```

Хотя написание самого бизнес-процесса важная вещь, тем не менее от него не будет толку без интеграции с Java-классами, в которых будут обрабатываться данные. Вызов происходит с помощью `Service Task` и для его корректной работы необходимо подготовить как бизнес-процесс, так и сам Java-класс.

Со стороны Activiti должен быть следующий код:

```
    <serviceTask id="javaService" name="Java service invoca-
tion" activ-
iti:class="org.activiti.examples.bpmn.servicetask.HelloWorld">
        <extensionElements>
            <activiti:field name="text" stringValue="Hello
World!" />
        </extensionElements>
    </serviceTask>
```

В теге `<serviceTask/>` заключена вся логика вызова определенных Java-классов. В атрибутах `id` и `name` прописываются идентификатор и имя сервиса заданий, а в `activiti:class` прописывается путь к классу и сам класс, который мы хотим вызвать. В теге `<extensionElements/>` содержатся дополнительные выражения, которые мы хотим передать.

Вызываемый нами Java-класс будет выглядеть следующим образом:

```
public class HelloWorld implements JavaDelegate {
    private Expression text;
    public void execute(DelegateExecution execution) {
        System.out.println(text);
    }
}
```

Вызываемый класс должен реализовывать интерфейс JavaDelegate, в котором объявлен всего один метод `execute(DelegateExecution execution)` [6]. В переменную типа `Expression` приходит то значение, которое мы поставили в бизнес-процессе. Стоит отметить, что названия переменных должны обязательно совпадать.

Кроме `Service Task` еще существует `Task Listener`, который используется для вызова класса при некотором событии, связанном с задачей (Task):

```
    <userTask id="myTask" name="My Task" >
        <extensionElements>
```

```
            <activiti:taskListener event="create"
class="org.activiti.MyTaskCreateListener" />
        </extensionElements>
    </userTask>
```

XML-код `Task Listener` несколько отличается от кода `Service Task`. Например, название вызываемого класса находится в теге `<extensionElements/>` и является обязательным. Еще один атрибут — event, так же является обязательным. События могут быть:

- `create` (создание);
- `assignment` (назначение кого-либо исполнителем);
- `complete` (совершение);
- `delete` (удаление).

Класс, в свою очередь, обязательно должен реализовывать интерфейс `TaskListener`, в котором объявлен один метод `notify(DelegateTask delegateTask)`:

```
    public class MyTaskCreateListener implements TaskListener
{
        public void notify(DelegateTask delegateTask) {
            // Custom logic goes here
        }
    }
```

Как видим из примеров, вызов и работа с Java-классами довольно проста. Так же просто и работать с уже готовыми `Activiti`-сервисами, наиболее популярными среди которых являются `HistoryService`, `IdentityService` и `TaskService`.

`HistoryService` используется для ведения истории процесса `Activiti`. `IdentityService` отвечает за пользователей и их группы. В Activiti пользователи заводятся следующим образом: вызывается два метода `IdentityService` — первый метод всего лишь создает транзитивный объект типа User, это метод `newUser(String userId)`, и метод `saveUser(User user)`, который сохраняет данного пользователя. Каждому объекту `User` можно присвоить свой уникальный номер, электронную почту, имя, фамилию, пароль и изображение. Аналогичным образом создаются и объекты типа `Group`, у которых есть уникальный номер, название и тип. После создания группы, в нее можно добавить пользователя методом `createMembership(String userId, String groupId)`, параметрами которого являются уникальный номер пользователя и ID группы, в которую необходимо поместить пользователя. Анало-

гично работает и удаление из группы. Еще одним важным методом сервиса является `checkPassword(String userId, String password)`. Данная булевая функция сравнивает пароль пользователя с передаваемым значением `password` и возвращает `true` в случае совпадения и `false` — в обратном.

Самым крупным сервисом есть `TaskService`. С помощью его методов можно добавлять и удалять комментарии, создавать и удалять задачи, прикреплять к ним документы, делегировать, выставлять приоритет и другое. Важно помнить, что всегда стоит вызывать метод `void saveTask(Task task)` для сохранения внесенных вами изменений.

Одним из видов связи Activiti и Spring является то, что в сущности, созданные ORM фреймворком Hibernate, можно добавлять ID любого процесса Activivi, тем самым позволяя разделить данные бизнес-процесса и программы, что является возможностью безболезненно править сущности, без потери данных. Например, можно создать свою сущность `CompletedTask`, у которой будут поля — `ID, processActivitiID, Date, Description`. При возникновении необходимости удалить поле Description, можно не бояться, что пропадут данные об исполнителях, комментариях уже существующих задач, поскольку они хранятся в таблицах, созданных фреймворком Activiti.

Ярким примером использования нотации BPMN и Activiti является один из крупных проектов реализации электронного документооборота Украины — iGov, где на практике применяются и реализовываются данные технологии. При первом знакомстве с данным проектом поначалу может показаться очень сложно понять принципы работы, во многом из-за небольшого количества и сложности информации. Однако изучив более подробно бизнес-процессы и применяя их на практике, можно осознать, насколько сильно они упрощают построение системы.

Выводы. Без использования бизнес-процессов, рабочая система будет усложнена, так как процессы переходов от одного события к следующему придется реализовывать программным методом, вследствии чего код будет разрастаться и со временем его будет тяжело поддерживать. Область применения бизнес-процессов не ограничена, но в первую очередь, они значительным образом облегчат функционал создания документооборота, в котором нужны и переходы на правильный шаг в зависимости от полученных данных, и создание сети пользователей, и конечно же, задач. Спецификация BPMN совсем не сложна в освоении, и при этом дает неограниченный потенциал для разработок. Тем не менее, справедливо и то, что для небольших проектов, которые бу-

дут пользоваться малой частью возможностей бизнес-процессов, например, создания пользователей, проверкой пароля на правильность и перенаправлением на той или иной шаг, такое использование бизнес-процессов избыточное и лишнее.

ЛИТЕРАТУРА

1. Spring [Электронный ресурс]. Режим доступа: https://spring.io/

2. BPM in Google Trends [Электронный ресурс].
Режим доступа: https://trends.google.com.ua/trends/explore?q=BPM

3. Activiti BPM Software Home [Электронный ресурс].
Режим доступа: https://www.activiti.org/

4. Activiti 5.x Business Process Management Beginner's Guide, Dr. Zakir Laliwala, Irshad Mansuri. Published by Packt Publishing Ltd. Livery Place 35 Livery Street Birmingham B3 2PB, UK, 2014.

5. Getting started with Activiti and Spring Boot [Электронный ресурс].
Режим доступа:
https://spring.io/blog/2015/03/08/getting-started-with-activiti-and-spring-boot

6. Activiti Java Documentation [Электронный ресурс]. Режим доступа:
https://www.activiti.org/javadocs/6.latest/

REFERENCES

1. Spring [Electronic resource]. Access mode: https://spring.io/

2. BPM in Google Trends [Electronic resource].
Access mode: https://trends.google.com.ua/trends/explore?q=BPM

3. Activiti BPM Software Home [Electronic resource].
Access mode: https://www.activiti.org/

4. Activiti 5.x Business Process Management Beginner's Guide, Dr. Zakir Laliwala, Irshad Mansuri. Published by Packt Publishing Ltd. Livery Place 35 Livery Street Birmingham B3 2PB, UK, 2014.

5. Getting started with Activiti and Spring Boot [Electronic resource].
Access mode: https://spring.io/blog/2015/03/08/getting-started-with-activiti-and-spring-boot

6. Activiti Java Documentation [Electronic resource].
Access mode: https://www.activiti.org/javadocs/6.latest/