RunaWFE. Структура файла-архива бизнес-процесса

Версия 3.0

Введение

RunaWFE - открытая, масштабируемая, ориентированная на конечного пользователя система управления бизнес-процессами для средних и крупных предприятий, основанная на популярном workflow ядре JBOSS-jBPM.

Бизнес-процессы в системе RunaWFE

Язык определения бизнес-процессов в системе RunaWFE основан на языке jPDL, (проект JBOSS jBPM). В языке определения бизнес-процессов системы RunaWFE есть несколько расширений jPDL. Некоторые возможности jPDL (например, подгрузка классов через файл-архив бизнес-процесса) пока системой RunaWFE не поддерживаются. Описание языка jPDL можно посмотреть по следующему URL: http://www.jbpm.org/jpdl.html [1].

В настоящем документе описывается язык определения бизнес-процессов системы RunaWFE.

Команда export редактора бизнес-процессов помещает описание бизнес-процесса в jar-архив с расширением .par.

Замечание. Файл должен быть jar-файлом без компрессии. Файл можно создать, например, командой «jar cvf0 <имя архива>.jar .».

Описание бизнес-процесса, помещенное в файл-архив, представляет собой набор XML-файлов и файлов, описывающих использующиеся в бизнес-процессе формы. Также архив может содержать графические файлы – графическое изображение графа бизнес-процесса и иконку бизнес-процесса.

При помощи интерфейса системы RunaWFE можно загрузить разработанный бизнес-процесс в систему.

После того, как бизнес-процесс загружен в систему, он появляется в списке бизнес-процессов, на него можно давать права и запускать на выполнение.

Структура файла-архива бизнес-процесса

Файл с расширением .par

- processdefinition.xml
- forms.xml
- variables.xml
- graph.gif
- start.png
- Файлы форм
- Файлы валидации форм

Граф бизнес-процесса и исполнители (перспектива потока управления и перспектива ресурсов) описываются в файле processdefinition.xml. Файл forms.xml, содержит список соответствующих узлам-Действиям форм. Файл graph.gif, содержит графическое изображение графа бизнес-процесса. Описания используемых в

бизнес-процессе форм находятся в файлах, имена которых определены в файле forms.xml. В файле variables.xml описаны переменные бизнес-процесса и их типы. В файлах валидации форм описаны проверки значения переменных, выполняющиеся при выполнении соответствующего задания.

Версии бизнес-процессов

Версионный механизм основан на следующих принципах:

- При загрузке новой версии определения бизнес-процесса в систему данные, соответствующие новой версии, запоминаются в базе данных. Этой версии присваивается следующий по порядку номер версии. Данные, соответствующие предыдущим версиям при этом не удаляются.
- Система рассматривает определения бизнес-процессов как разные версии одного процесса, если у них совпадает имя процесса.
- Экземпляр бизнес-процесса все время соответствует определению бизнес-процесса, обладавшему наибольшим номером версии на момент запуска этого экземпляра бизнес-процесса. Если во время выполнения этого экземпляра в систему будут загружены следующие версии определения бизнес-процесса, они уже не окажут влияния на его выполнение.

Описание элементов языка определения бизнес-процессов, используемого в системе RunaWFE

Описание файла processdefinition.xml и используемых в нем тегов

В системе RunaWFE предусмотрены такие понятия, как группы пользователей и функции над оргструктурой. Для того, чтобы иметь возможность с ними работать, надо определять элементы swimlane специальным образом. Это подробно описано в следующем разделе.

Использование ролей (элемента swimlane) в системе RunaWFE

Swimlane (роль-Дорожка) –представляет собой специальный тип переменных бизнес-процесса. Используется для определения Пользователей, которые могут выполнить определенное Действие. Роль-Дорожка ставится в соответствие узлу-Действию. В RunaWFE до начала исполнения Действия роли-Дорожке должен быть поставлен в соответствие инициализатор, который возвращает пользователя, либо группу пользователей. Инициализация роли-Дорожки состоит в том, что ей (как переменной) присваивается ID пользователя или группы пользователей. Если роль-Дорожка еще не проинициализирована, то ее инициализация происходит в момент прихода управления в данный узел-Действие. Если роль-Дорожка проинициализирована группой пользователей, то после того, как Действие выполнено, производится доинициализация роли-Дорожки: в роль-Дорожку вместо ID группы помещается ID того пользователя, который выполнил данное Действие.

Далее по приходу в Действие, где исполнитель соответствует этой роли-Дорожке просходит проверка нужно ли переинициализировать роль. Этот параметр можно выбрать в настройках роли на узле-Действии процесса в графическом редакторе при создании\редактировании процесса. Если стоит признак, что необходима переинициализация, то роли-Дорожке снова будут поставлены в соотвествие все члены группы, обозначенной в начальном инициализоторе роли-Дорожки и соответствующее Действие будет назначено всем членам группы. Первый выполнивший Действие снова доинициализирует роль-Дорожку своим ID пользователя. Если признак переинициализации отсутствует, то Действие будет назначено только тому пользователю, чьим ID была доинициализирована роль-Дорожка в предыдущий раз.

Опишем понятие инициализации более подробно:

С понятием роль-Дорожка в системе RunaWFE связан алгоритм назначения заданий:

В момент, когда управление попадает в данный узел_Действие, происходит следующее:

- Если роль-Дорожка проинициализирована группой пользователей, то задания получат все пользователи, входящие в группу, однако только тот пользователь, который первым выполнит это задание, заново проинициализирует роль-Дорожку своим ID.
- Если роль-Дорожка уже проинициализирована, и не выставлен признак переинициализации, то задание будет направлено только тому пользователю, ID которого она проинициализирована.
- Если роль-Дорожка уже проинициализирована, и выставлен признак переинициализации, то роль дорожка будет снова проинициализирована группой пользователей, как в начале.

Если роль-Дорожка не определяется явно как переменная (формой, ботом или в start-state), то ее определение в файле processdefinition.xml обязательно должно содержать тег delegation, в котором должен быть указан класс, наследующий интерфейс org.jbpm.delegation.AssignmentHandler (как правило, это класс ru.runa.wf.jBPM.delegation.assignment.AssignmentHandler) и строку инициализации роли-Дорожки.

Строка инициализации роли-Дорожки должна представлять собой следующее:

<Класс специального вида>(<параметр>, <параметр>, …)

В настоящее время нами разработаны следующие классы, которые можно употреблять в данном выражении:

- ru.runa.af.organizationfunction.ExecutorByNameFunction – в параметр надо передать имя пользователя или группы, инициализатор будет возвращать ID этого пользователя или ID группы
- ru.runa.af.organizationfunction.DemoChiefFunction – в параметр надо передать имя пользователя, инициализатор будет возвращать руководителя этого пользователя (класс разработан для Демо-примеров)

Каждый параметр представляет собой либо строку, либо $(<имя переменной бизнес процесса>). В случае переменной бизнес-процесса классу передается значение этой переменной, в случае обычной строки – классу передается эта строка.

Замечание. В самом jPdl строка инициализации роли-Дорожки не специфицирована.

Описание структуры файла

Файл processdefinition.xml представляет собой XML-документ, содержащий тег process-definition.

Тег process-definition описан ниже в данном разделе.

Пример файла processdefinition.xml:

<?xml version="1.0" encoding="UTF-8"?>

<process-definition name="MainProcess" xmlns="urn:jbpm.org:jpdl-3.2">

<description><![CDATA[MainProcess for subprocess demonstration (no start form proveded)]]></description>

<swimlane name="a_role">

<assignment class="ru.runa.wf.jbpm.delegation.assignment.AssignmentHandler" config-type="configuration-property"><![CDATA[]]></assignment>

</swimlane>

<start-state name="start">

<task name="start" swimlane="a_role"/>

<transition name="tr1" to="before subprocess"/>

</start-state>

<task-node name="before subprocess">

<description><![CDATA[before subprocess]]></description>

<task name="before subprocess" swimlane="a_role"/>

```
<transition name="tr1" to="process-state"/>
</task-node>
<task-node name="after subprocess">
<description><![CDATA[after subprocess]]></description>
<task name="after subprocess" swimlane="a_role"/>
<transition name="tr1" to="end"/>
</task-node>
<event type="subprocess-created">
<action class="ru.runa.wf.jbpm.delegation.action.SetSubProcessPermissionsActionHandler" config-type="configuration-property"><![CDATA[]]> <nowiki></action>
</event>
<process-state name="process-state">
<transition name="tr1" to="after subprocess"/>
<sub-process binding="late" name="SubProcess"/>
<variable access="read" mapped-name="role1" name="a_role"/>
<variable access="read,write" mapped-name="n" name="number"/>
<variable access="read,write" mapped-name="d" name="date"/>
</process-state>
<end-state name="end"/>
</process-definition>
```

Описание тега process-definition

Первым элементом тега является необязательный элемент description. Далее следует блок описания ролей-дорожек (набор тегов swimlane). Далее следует блок описаний типов (набор тегов type). Далее следует начальный узел-Действие в графе бизнес-процесса (тег start-state). Этот тег должен обязательно содержать переход (тег transition) в следующий узел бизнес-процесса. Также этот тег содержит параметр swimlane. Параметру присваивается имя роли-Дорожки, которая будет инициализирована пользователем, запустившим экземпляр бизнес процесса.

Далее идет список тегов, соответствующих обычным узлам бизнес-процесса. Список типов обычных узлов бизнес-процесса:

- state (узел-Действие)
- decision (исключающий выбор)
- fork (параллельное расщепление)
- join (синхронизация)
- process-state (подпроцесс)

Каждый узел-Действие должен содержать тег assignment с параметром swimlane – значение этого параметра будет определять Исполнителя, которому будет направлено задание после прихода управления в данный узел-Действие бизнес-процесса. Каждый узел должен содержать одну или несколько ссылок на «следующие» узлы бизнес-процесса.

Следующим вложенным тегом тега бизнес-процесса должен быть тег, соответствующий точке окончания бизнес-процесса (тег end-state). После прихода управления в этот узел бизнес-процесс завершается.

Замечание. В отличие от jPDL, язык описания бизнес-процессов системы RunaWFE игнорирует тег type. Также язык игнорирует тег action, вложенный в тег process-definition.

Описание элемента swimlane

Swimlane (роль-Дорожка) –представляет собой специальный тип переменных бизнес-процесса. Используется для определения Пользователей, которые могут выполнить определенное Действие. Роль-Дорожка ставится в соответствие узлу-Действию.

Описание элемента start-state

Элемент start-state соответствует точке старта процесса. В описании бизнес процесса этот элемент должен присутствовать в единственном экземпляре. В отличие от обычных узлов графа бизнес-процесса, start-state содержит параметр swimlane. Значением параметра является имя роли-Дорожки, которая будет инициализирована ID пользователя, который запустил бизнес-процесс.

К start-state может быть присоединена графическая форма. В этом случае она сразу будет показана после выполнения команды «запустить процесс». Эта форма используется для ввода начальных данных бизнес-процесса. До нажатия на кнопку «выполнить» этой формы бизнес-процесс еще реально не будет запущен, то есть, если вообще не заполнить входящую форму (например, закрыть web-браузер), то бизнес процесс не стартует.

Описание элемента state

Элемент state соответствует узлу-Действию (activity в терминах диаграммы деятельности языка UML).

Описание вложенного элемента assignment

Элемент используется для определения Пользователей, которые могут выполнить определенное Действие. При помощи параметра swimlane определенная Роль-Дорожка ставится в соответствие данному узлу-Действию. Если к моменту выполнения Действия эта роль-Дорожка еще не была проинициализирована, то она инициализируется ID того пользователя, который выполнил данное действие.

Вложенный элемент action

К элементу state могут быть «присоединены» actions (реализованы java-классами специального вида). Соответствующий код этих классов будет выполнен в случае, если произойдут некоторые event'ы, такие, как

- Приход точки управления в узел-Действие
- Уход точки управления из узла-Действия
- и т.д.

Подробно элемент action описан ниже в данном документе.

Вложенный элемент transition

Элемент transition указывает на следующий узел, в который перейдет точка управления.

Подробно элемент transition описан ниже в данном документе.

Описание элемента milestone

Элемент milestone соответствует паттерну milestone. Текущая версия нашего языка этот элемент не поддерживает.

Описание элемента process-state

Элемент process-state запускает подпроцесс. При этом базовый процесс ждет в этом узле окончания подпроцеса.

Описание элемента decision

Элемент decision соответствует маршрутному узлу «Исключающий выбор». «Выбирает» на основании текущих значений переменных бизненс-процесса один из нескольких возможных исходящих переходов. Правила, по которым осуществляется выбор перехода, задаются при помощи тега delegation. Разработанный нами класс ru.runa.wf.jbpm.delegation.decision.BSFDecisionHandler осуществляет выбор перехода на основании переданного ему в теле тега BeanShell скрипта (см. http:/ / www. beanshell. org/ intro. html [2]). В теле конфигурации доступны все переменные бизнес-процесса (Желательно при их использовании явно приводить их тип). Скрипт должен возвращать значение типа String, совпадающее с именем одного из исходящих переходов.

Пример использования тега:

```
<decision name="check business trip type">
<delegation class="ru.runa.wf.jbpm.delegation.decision.BSFDecisionHandler">
<![CDATA[
if( (String) businessTripType.equals("local"))
return "local";
else
return "toAnotherRegion";
]]>
</delegation>
<transition name="local" to="done"/>
<transition name="toAnotherRegion" to="Make an order"/>
</decision>
```

Описание элемента fork

Элемент fork соответствует маршрутному узлу «Параллельное расщепление». Для пришедшей в узел точки управления генерируются точки управления для всех исходящих переходов. Все сгенерированные точки управления далее выполняются параллельно. Требует «закрывающего» элемента join, «собирающего» все порожденные fork'ом точки управления, вместе с которым образует так называемый параллельный блок: Область в бизнес-процессе, у которой есть одна точка входа и одна точка выхода (вход-выход точек управления через «боковые стороны» параллельного блока запрещен).

Замечание. Поведение элемента fork можно переопределить через delegation. Для этого существует специальный интерфейс ForkHandler.

Описание элемента join

Элемент join соответствует маршрутному узлу «Синхронизация». Имеет только один исходящий переход. Этот узел «собирает» все порожденные соответствующим fork'ом точки управления. После того, как все точки управления пришли в join, из него выходит единственная точка управления, которая перемещается в следующий узел.

Замечание. Поведение элемента join можно переопределить через delegation. Для этого существует специальный интерфейс JoinHandler.

Описание элемента end-state

Элемент end-state соответствует точке окончания процесса. В описании бизнес процесса этот элемент должен присутствовать в единственном экземпляре. В момент прихода управления в этот узел бизнес-процесс полностью завершается.

Описание элемента transition

Элемент transition определяет переход между узлами бизнес-процесса.

Описание элемента action

Элемент action определяет java код, который будет выполнен ядром WF-системы в случае возникновения тех или иных событий (events) во время выполнения бизнес-процесса.

Поведение элемента action можно определить через delegation. Для этого существует специальный интерфейс ActionHandler.

Описание элемента delegation

Delegation – специальный механизм, при помощи которого разработчик бизнес-процесса может включать в бизнес-процесс свои собственные Java классы. Для загрузки этих классов в ядро в системе предусмотрен специальный class loader.

В зависимости от того, внутри какого тега использован delegation, прилагаемый Java-класс должен реализовывать определенный интерфейс. Например, в случае тега action, это интерфейс ActionHandler, в случае тега decision, это интерфейс DecisionHandler и т.д. Также delegation-класс всегда реализует интерфейс Configurable.

Delegation задается при помощи следующих составляющих:

1. Имя используемого класса – атрибут class (обязательно)
2. Конфигурация для delegation – #PCDATA в теле тега (не обязательно)

Описание файла forms.xml

В JBOSS JBPM структура файла forms.xml определена неформально и этот файл не является обязательным. В RunaWFE структура файла forms.xml определена строго и файл является обязательным. Файл состоит из единственного тега forms. Внутри тега forms находится набор тегов form. Каждый тег соответствует узлу, которому соответствует графическая форма, или в котором присваиваются значения переменным бизнес-процесса.

Файл forms.xml состоит из единственного тега forms. Внутри тега forms находится набор тегов form. Каждый тег соответствует узлу, которому соответствует графическая форма, или в котором присваиваются значения переменным бизнес-процесса.

У тега form есть три обязательных атрибута

- state – название узла бизнес-процесса
- file – имя файла, соответствующего графической форме, которая будет показана в проигрывателе форм для задания из данного узла.
- type – тип формы ("html" (форма с vartag), "ftl" (форма с freemarker тегами), "xsn" (infopath форма))

Полностью XML-схема, определяющая forms.xml находится в папке resource в дистрибутиве системы.

Описание файла определения форм (.html, .ftl файлы)

Каждый файл содержит описание формы на языке HTML, расширенном при помощи дополнительного тега customtag (.html формы) иди freemarker тегов. Совместное использование и customtag и freemarker тегов в одной и той же форме невозможно.

Тег customtag содержит следующие атрибуты

- var – имя переменной бизнес-процесса
- delegation – имя класса, который будет использован при работе с переменной через графическую форму (класс должен реализовывать интерфейс VarTag)

Примеры построения файлов-определений бизнес-процессов.

HelloWorld процесс.

Построим простейший бизнес-процесс. Процесс будет заключаться в следующем:

После запуска бизнес-процесса на экране появится форма HelloWorld, после нажатия кнопки “Выполнить” в этой форме, процесс завершится.

Этот процесс будет состоять из трех узлов-Действий:

- Начальный узел-Действие, совпадающий с точкой начала бизнес-процесса
- Точка окончания бизнес-процесса

Файл processdefinition.xml будет выглядеть следующим образом:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Начало тега process-definition -->
<process-definition name="Hello World" xmlns="urn:jbpm.org:jpdl-3.2">
<!-- Определение роли-Дорожки -->
<swimlane name="requester" />
<!-- Точка начала бизнес-процесса и ввода начальных данных -->
<start-state name="Hello World state" swimlane="requester">
<!-- Переход в следующий узел -->
<transition to="done"/>
</start-state>
<!-- Точка завершения бизнес-процесса -->
<end-state name="done" />
```

<!-- Завершение тега process-definition -->

</process-definition>

Файл forms.xml будет выглядеть следующим образом:

<?xml version="1.0"?>

<forms xmlns="http://runa.ru/xml" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://runa.ru/xml forms.xsd">

<!-- Тег связывает узел-Действие с графической формой -->

<form state="Hello World state" file="forms/HelloWorld.form" type="html">

</form>

</forms>

В файл graph.gif запишем следующее изображение:

Файл HelloWorld.form может быть, например, следующим:

Hello World!

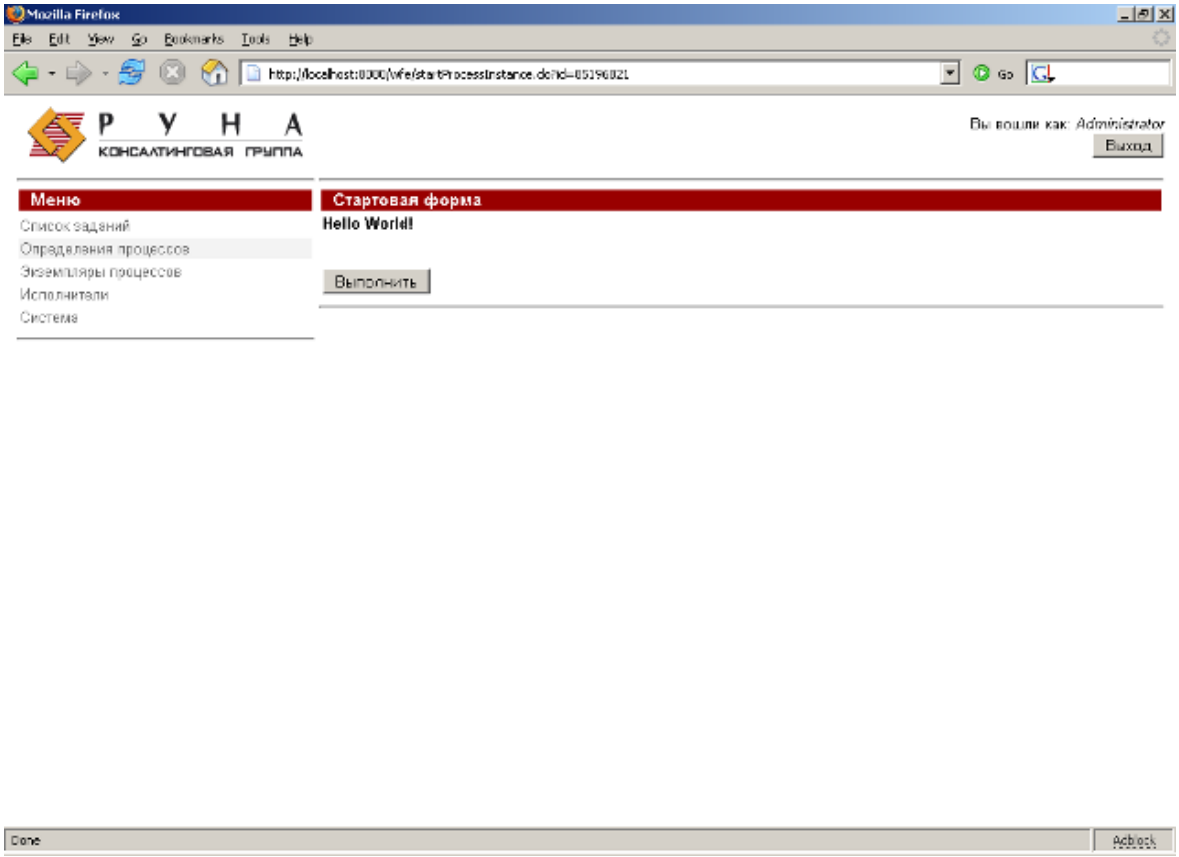
Структура архива HelloWorld.par будет следующей:

Файл HelloWorld.par

- processdefinition.xml
- forms.xml
- graph.gif
- HelloWorld

Процесс можно загружать в систему.

При выполнении форма должна выглядеть так:



Процесс сверхурочные.

Краткое описание процесса:

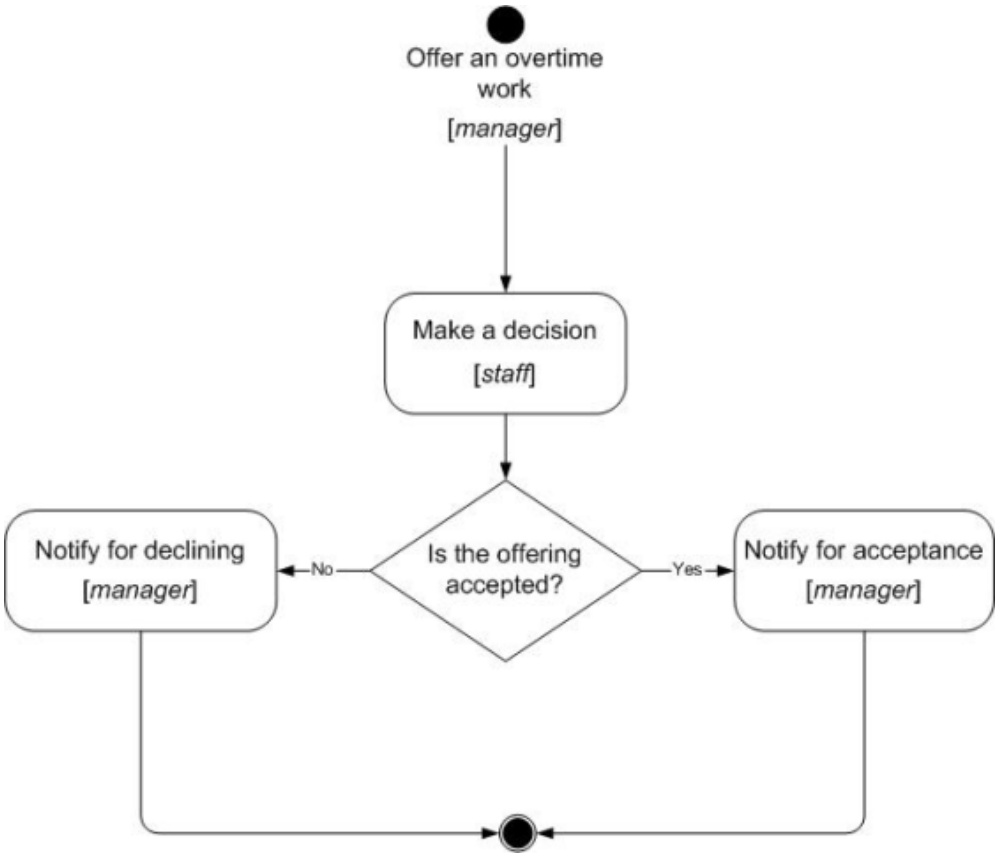
Руководитель предлагает сотруднику выйти на работу сверхурочно – сотрудник соглашается или отказывается. Руководитель получает уведомление соответственно о согласии или об отказе.

Предположим, что все руководители являются членами группы manager, а все сотрудники – членами группы staff.

Проектирование бизнес-процесса

Перспектива управления потоком

Перспектива управления потоком будет соответствовать следующему графу:



Перспектива данных

Введем следующие переменные бизнес-процесса:

Переменная	**Описание переменной**	**Тип**	**Узел, в котором переменная инициализируется**
staff	ID работника	ID	Offer an overtime work
since	Дата-время с…	Дата-время	Offer an overtime work
till	Дата-время по …	Дата-время	Offer an overtime work
reason	Причина	Строка	Offer an overtime work
comment	Комментарий	Текст	Offer an overtime work
staff person decision	Решение работника	Логический	Make a decision
staff person comment	Комментарий работника	Текст	Make a decision

Перспектива Ресурсов

Введем следующие роли-Дорожки:

- manager - руководитель
- staff person - работник

Инициализация ролей-Дорожек:

Роль-Дорожка	**Как инициализируется**
manager	Тот, кто запустил бизнес-процесс. Предполагается, что права на запуск данного бизнес-процесса есть только у членов группы manager (руководителей)
staff	Члены группы staff. Предполагается, что в эту группу входят все работники предприятия

Таблица соответствия – в каких узлах какие роли-Дорожки используются:

Узел-Действие	**Роль-Дорожка**
Offer an overtime work	manager
Make a decision	staff
Notify for declining	manager
Notify for acceptance	manager

Перспектива операций

Обмен данными в этом бизнес-процессе происходит только через графические формы. Соответствие переменных и форм выписано в таблице переменных бизнес-процесса. Тип HTML элемента определяется типом переменной бизнес-процеса во всех случаях, кроме переменной staff. Ее значение определяется Choice'ом, содержащим всех членов группы staff.

Файл-архив бизнес-процесса

Файл processdefinition.xml

<?xml version="1.0" encoding="UTF-8"?>

<process-definition name="Overtime Work" xmlns="urn:jbpm.org:jpdl-3.2">

<description><![CDATA[Participants of this process are members of manager and staff groups]]></description>

<swimlane name="staff">

<assignment class="ru.runa.wf.jbpm.delegation.assignment.AssignmentHandler" config-type="configuration-property"><![CDATA[]]></assignment>

</swimlane>

<swimlane name="manager">

<assignment class="ru.runa.wf.jbpm.delegation.assignment.AssignmentHandler" config-type="configuration-property"><![CDATA[]]></assignment>

</swimlane>

<start-state name="Offer an overtime work">

<task name="Offer an overtime work" swimlane="manager"/>

<transition name="tr1" to="Make a decision"/>

</start-state>

<decision name="Is accepted?">

<handler class="ru.runa.wf.jbpm.delegation.decision.BSFDecisionHandler" config-type="configuration-property">
<![CDATA[if(Boolean.valueOf(staffPersonDecision).booleanValue()) return "accepted"; else return "rejected";]]></handler>
<transition name="accepted" to="Notify of acceptance"/>
<transition name="rejected" to="Notify of rejection"/>
</decision>
<task-node name="Make a decision">
<description><![CDATA[Accept or decline the offering]]></description>
<task name="Make a decision" swimlane="staff"/>
<transition name="tr1" to="Is accepted?"/>
</task-node>
<task-node name="Notify of acceptance">
<description><![CDATA[Notify that an overtime work is accepted]]></description>
<task name="Notify of acceptance" swimlane="manager"/>
<transition name="tr1" to="end"/>
</task-node>
<task-node name="Notify of rejection">
<description><![CDATA[Notify that an overtime work is rejected]]></description>
<task name="Notify of rejection" swimlane="manager"/>
<transition name="tr1" to="end"/>
</task-node>
<end-state name="end"/>
</process-definition>

Файл forms.xml

<?xml version="1.0" encoding="UTF-8"?>
<forms xmlns="http://runa.ru/xml" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://runa.ru/xml forms.xsd">
<form file="Offer an overtime work" jsValidation="false" state="Offer an overtime work" type="html" validationFile="Offer an overtime work.validation.xml"/>
<form file="Make a decision" jsValidation="false" state="Make a decision" type="html" validationFile="Make a decision.validation.xml"/>
<form file="Notify of acceptance" jsValidation="false" state="Notify of acceptance" type="html" validationFile="Notify of acceptance.validation.xml"/>
<form file="Notify of rejection" jsValidation="false" state="Notify of rejection" type="html" validationFile="Notify of rejection.validation.xml"/>
</forms>

Файл variables.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<variables xmlns="http://runa.ru/xml" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://runa.ru/xml variables.xsd">
<variable format="org.jbpm.web.formgen.format.DefaultFormat" name="staff"/>
<variable format="ru.runa.wf.web.forms.format.DateTimeFormat" name="since"/>
<variable format="ru.runa.wf.web.forms.format.DateTimeFormat" name="till"/>
<variable format="org.jbpm.web.formgen.format.DefaultFormat" name="reason"/>
<variable format="org.jbpm.web.formgen.format.DefaultFormat" name="comment"/>
<variable format="org.jbpm.web.formgen.format.DefaultFormat" name="staff person comment"/>
<variable format="ru.runa.wf.web.forms.format.BooleanFormat" name="staffPersonDecision"/>
</variables>
```

Файл graph.gif

Содержание файла соответствует рисунку, приведенному в разделе «Перспектива управления полтоком»

Файлы форм

Файл OfferAnOvertimeWork:

```
<link href="form.css" type="text/css" rel="stylesheet" />
<div class="form-container">
<p class="legend"><strong>Offer an overtime work</strong></p>
<div><label for="staff">Employee <em>*</em></label>
<!-- Специальный тег, расширяющий HTML, выдает на экран choice,
содержащий список членов группы, название которой передается
в переменной var=… . Возвращает ID выбранного члена группы.
В теге используется механизм delegation -->
<customtag var="staff" delegation="ru.runa.wf.web.html.vartag.GroupMembersComboboxVarTag" /></div>
<div><label for="since">Since <em>*</em></label><em><font size="-1">
<!-- Специальный тег, расширяющий HTML, служит для работы с датами-->
<customtag var="since" delegation="ru.runa.wf.web.html.vartag.DateTimeInputVarTag" /> (dd.mm.yyyy)</font></em></div>
<div><label for="till">Till <em>*</em></label><em><font size="-1"><customtag var="till" delegation="ru.runa.wf.web.html.vartag.DateTimeInputVarTag" /> (dd.mm.yyyy)</font></em></div>
<div><label for="reason">Reason <em>*</em></label><input id="reason" name="reason" /></div>
<div><label for="comment">Comments</label><textarea id="comment" wrap="hard" name="comment"></textarea></div>
</div>
```

Аналогично данному файлу строятся файлы:

- MakeaDecision.form
- NotifyForAcceptance.form
- NotifyForDeclining.form

Файлы валидаций (проверок) для переменных форм

Файл Offer an overtime work.validation.xml

```
<validators>
<field name="since">
<field-validator type="required">
<message>Field is required</message>
</field-validator>
</field>
<field name="reason">
<field-validator type="stringlength">
<message>Length cannot be more than 100 symbols</message>
<param name="maxLength">100</param>
</field-validator>
<field-validator type="required">
<message>Value is required</message>
</field-validator>
</field>
<field name="staff">
<field-validator type="required">
<message>Field is required</message>
</field-validator>
</field>
<field name="comment">
<field-validator type="stringlength">
<message>Length cannot be more than 255 symbols</message>
<param name="maxLength">255</param>
</field-validator>
</field>
<field name="till">
<field-validator type="required">
<message>Field is required</message>
</field-validator>
</field>
<validator type="expression">
<message>Till should be later that since</message>
<param name="expression">till.getTime() > since.getTime()</param>
</validator>
</validators>
```

Аналогично строятся файлы валидации для остальных форм.

Структура архива

Файл overTimeDemo.par

- processdefinition.xml
- forms.xml
- graph.gif
- Файлы форм:
  - OfferAnOvertimeWork
  - MakeaDecision
  - NotifyForAcceptance
  - NotifyOfRejection
- Файлы валидации переменных форм:
  - Offer an overtime work.validation.xml
  - Make a decision.validation.xml
  - Notify of acceptance.validation.xml
  - Notify of rejection.validation.xml

Далее процесс можно загружать в систему.

Замечание. Дистрибутив системы также содержит другие демо-процессы:

- VacationDemo.par – отпуск
- BusinessTripDemo – командировка
- TimerDemo – пример использования таймера

Примечания

[1] http://www.jbpm.org/jpdl.html

[2] http://www.beanshell.org/intro.html