

kbeeworkflow

Tutorial.



kbee.workflow Tutorial

Document Properties

Project:

kbee.workflow

Document Title:

Tutorial

Version date:

15/04/07

Version number:

V003

Author:

Novamens



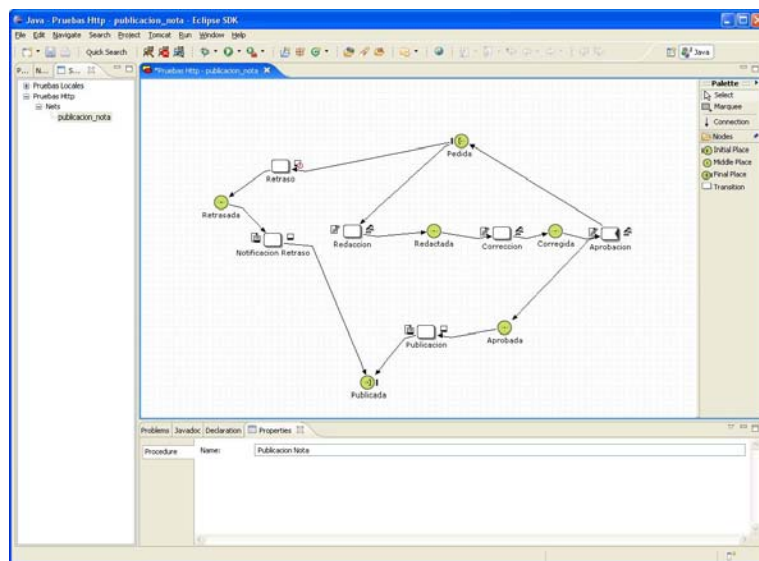
1	Introduction	4
1.1	Objetive	4
2	The Problem: publication of articles in a publishing media.....	5
2.1	Description.....	5
2.2	Analysis	5
3	Solution	7
3.1	Contexts	7
3.2	Tasks	8
3.3	Triggers	9
3.4	Class factory.....	11
4	Execution and Trials	12
4.1	Web Console.....	12
4.2	Process Triggering.....	13
4.3	WQL Monitoring.....	14

1 Introduction

1.1 Objective

The objective of the following tutorial is to provide a solution to a simple reference problem in which the use of kbee.workflow can be explained in detail.

The net explaining the complete specification of the solution procedure is included in the distribution under the name of 'publicación_nota'. In order to open and read the specification, developers should first open the corresponding view (Window -> Show View -> Others -> Workflow -> Servers) in Eclipse and then open said procedure.



2 The Problem: publication of articles in a publishing media

2.1 Description

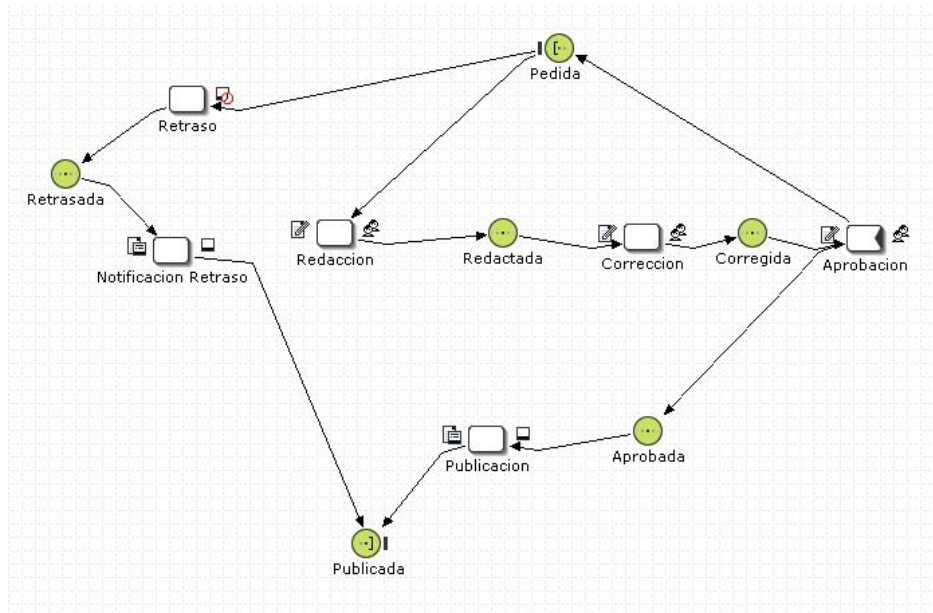
A publishing media needs to implement an editorial production cycle in order to publish articles:

- The process is triggered by the order of a person in charge requesting an article on a given subject.
- The writing of the article is carried out by a writer.
- After it, the article is corrected by one of the user members from the editors' staff (In this case, any of the users who are members of the editors' staff can perform the editing task)
- Finally, the article must be approved by a supervisor before it is published.
- An order will be rejected if no writer performs the task within a short period of time.

As a non-functional requirement, it is stated a web solution must be implemented.

2.2 Analysis

As part of the analysis and drawing of a problem, and even before its development, developers can draw and document the required procedure. The following format is completely independent from any technology and it is only shown as a suggestion and as a mechanism for the detailed specification of the reference problem:



2.2.1 Writing task

Definition: a writer develops an article according to given instructions.

Pre condition: beginning of process.

Trigger: manual. The enabled user indicates the beginning and ending of the task.

Actor: writer.

Post condition: written article.

Poscondición: nota redactada.



2.2.2 Editing Task

Definition: an editor performs orthographic and style corrections upon the writer's work.

Pre condition: written article.

Trigger: manual. The enabled user indicates the beginning and ending of the task.

Actor: editor.

Post condition: corrected article.

2.2.3 Approval Task

Definition: The user in charge approves or rejects the work.

Pre condition: corrected article.

Trigger: manual. The enabled user indicates the beginning and ending of the task.

Actor: supervisor.

Post condition: approved article ready for publication, or rejected article sent back to editor..

2.2.4 Publication Task

Definition: An automatic task publishes the approved article. (For example, the publication may generate an HTML version and a copy to a public portal).

Pre condition: approved article. The task is executed after the pre condition is satisfied.

Trigger: automatic. The enabled user indicates the beginning and ending of the task.

Actor: system.

Post condition: published article.

3 Solution

3.1 Contexts

Each token circulating through a process net contains a data structure referred to as context. The aim of a context is to include references to one or more objects of the domain where the process is executed. It also contains information only valid within the process context, such as priorities or specific instructions for the process or tasks.

3.1.1 Context definition

The context works as a black box for the server which can only be accessed using JXPath (Java + XPath). It allows the context implementation to be completely disconnected from the server.

The default implementation of a context included in the server is a XML DOM object. According to the reference problem, it should follow this schema:

```
<xs:schema xmlns:xs='http://www.w3.org/2001/XMLSchema'>
  <xs:element name="Nota">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="indicaciones"/>
        <xs:element ref="id"/>
        <xs:element ref="titulo"/>
        <xs:element ref="seccion"/>
        <xs:element ref="texto"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Although it is desired that the context contains only references, and that the persistence and management of objects (or documents) is sent to the applications tier (for example, a Cms), in the case of the reference problem the whole content is stored in the same context as a way of simplification.

3.1.2 Registering context

In order to register the specified structure of the context in the server, developers should follow these steps:

In the relational table **PETRI_CLASS** add a row with the name of the type of context to be created:

PTR_NAME
Nota

In the relational table **PETRI_DATA** add the following rows to describe the structure:



PTR_DATA	PTR_PATH	PTR_VARIABLE	PTR_INDEXABLE	PTR_CLASS	PTR_ORDER
Indicaciones	Context/*/indicaciones	1	1	Nota	1
Nota	Context/*/nota	1	1	Nota	2
Título	Context/*/titulo	1	1	Nota	3
Sección	Context/*/seccion	1	1	Nota	4
Texto	Context/*/texto	1	0	Nota	5

The meanings of the above columns are:

- **PTR_DATA**: unique identification of data
- **PTR_PATH**: the JXPath applied to the context to obtain the corresponding value.
- **PTR_VARIABLE**: variables of the context value through the process cycle.
- **PTR_INDEXABLE**: indexable data extends the native model and can be queried through WQL.

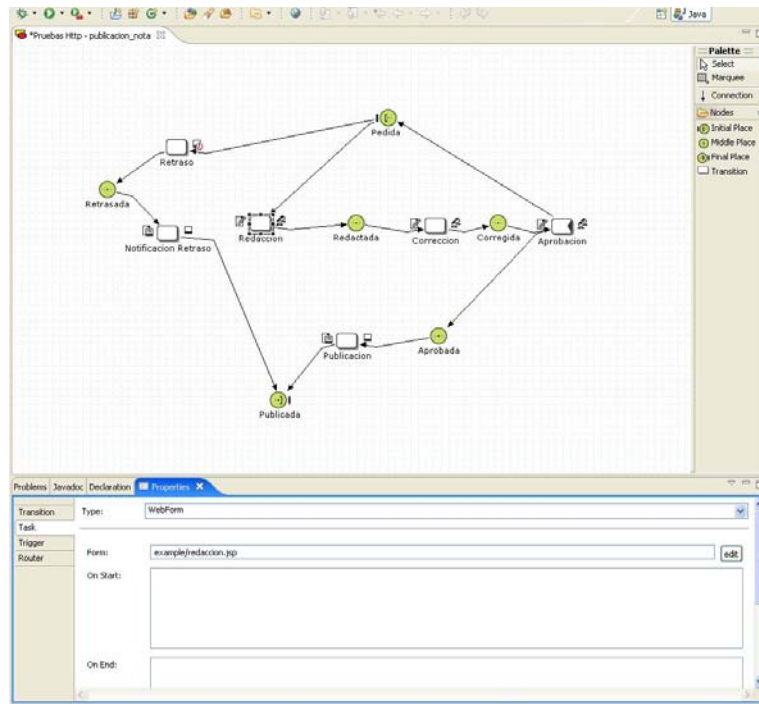
3.2 Tasks

3.2.1 Web Forms

These tasks can be performed by the user using a Web tool. The basic specification data is the web-tool URL. The server forwards the URL by only sending the identifier of the corresponding entity (activity or work item).

The web page receiving the forward from the server should perform the following steps:

1. Retrieving the data from the corresponding entity and context.
2. Creating a HTML frame of the task including the task name, the process or tasks instructions and the buttons related to the beginning or ending of the task.
3. Passing the parameters to the tool that will be definitely used to solve the task. In the case of the reference problem, an HTML editor is created for the writing, editing and approval tasks.



As an example, the sources of the writing, editing and approval tasks can be revised.

3.2.2 Java Script

These tasks allow developers to code scripts in order to solve automatic tasks. The script receives the context instance as a variable, and from that point on it should be able to access any domain data.

As an example of these, we can mention the publication name in the reference problem.

3.3 Triggers

3.3.1 Manual

A manual trigger can only be started by a user, e.g. by pressing a button.

The trigger specification should include the following data:

- **Group:** it refers to the 'main' or reference group upon which the access control is executed. Typically, verification is executed to check if a user belongs to the group.
- **Condition:** it refers to activation condition upon the context. A specified condition should be satisfied in order to fire the trigger. If no condition is specified the triggered is not fired.
- **Controller:** it refers to control strategy. In order to control the user's access, the strategy 'member' verifies if the user belongs to the group specified in the trigger. To define other controllers see 'Class Factory'.



Examples of manual triggers are the ones specified in the writing, editing and approval tasks of the reference problem.

It should be taken into account that the delivered security manager only functions as stub and that it should be changed in order to be integrated to a requiring security system (See 'SecurityManager' in 'Class Factory').

3.3.2 Automatic

An automatic trigger is fired after the pre-condition of a task is satisfied.

The trigger specification should include the following data:

- **Condition:** it refers to the activation condition in the context. A specified condition should be satisfied in order to fire the trigger. If no condition is specified the triggered is not fired.
- **Signaller:** a signaller fires events for the re-evaluation of a specified condition. Examples of signaller are: a timer or a JMS queue subscriber. In the first case, the condition will be re-evaluated in each time-cycle determined by the timer. In the second case, the condition is re-evaluated after the reception of a JMS message. See point 'Class Factory' for signallers' definitions.

As an example of automatic trigger we can mention the publication task of the reference problem.

3.3.3 Timeout

A timeout trigger is fired after a specific period of time. Its definition only includes the specified time.



3.4 Class factory

Every class in the application is instanced through a class factory. This entity keeps a record of classes in the system under a unique alias.

The corresponding implementation as well as the selection of instances for each part and the initial value of attributes is indicated in each class record.

The files containing the records of every class used in the application are located in the directory *WEB-INF/workflow-config/engine/classes* in the server's web application.

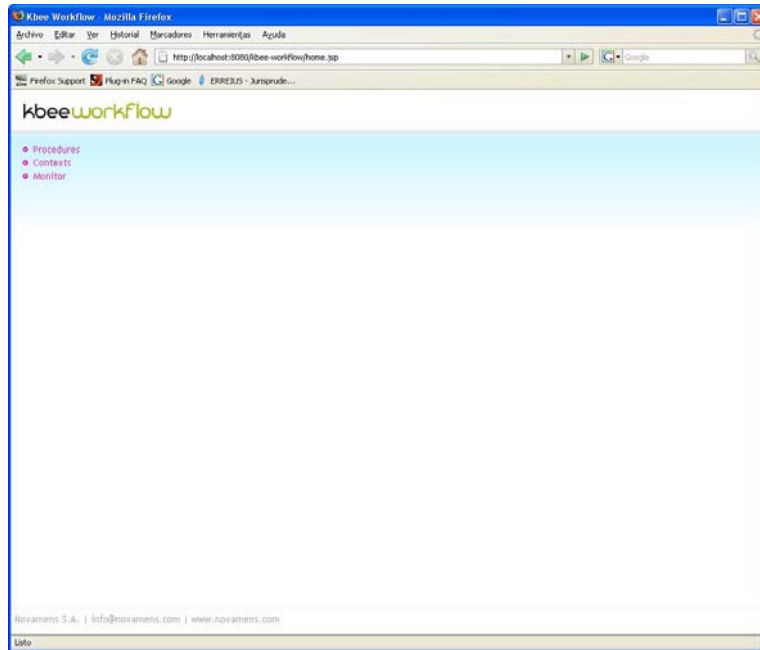
Each class implementation and parameter can be changed by altering these files. A recommendable alternative is to create new records in the file *WEB-INF/workflow-config/classes/classes.xml*. The records in this file can re-define all the records in the application.



4 Execution and Trials

4.1 Web Console

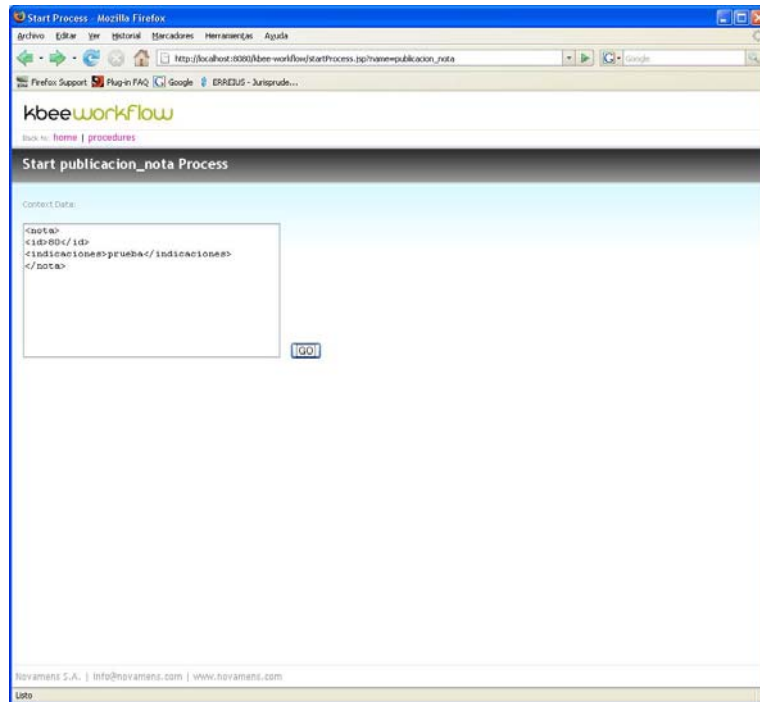
The web console included in the server enables the user to query procedures, trigger processes, query defined contexts and execute WQL queries for monitoring.



4.2 Process Triggering

The console comprises of a generic tool for process triggering in which developers need to enter the complete context data for the new process.

The following figure shows how to trigger a process included in the publication of articles procedure (*Home->Procedures->publicación_notas->Stara Process*):





4.3 WQL Monitoring

The console also comprises of a monitoring tool used to write WQL sentences. The sentences included in this tool are shown as an example:

