Runa WFE 2.1

Administration Guide

Legal Notice

This program is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; version 2.1 of the License. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.

Table of Contents

Legal Notice 2
Introduction 5
Installation 6
Binary distribution 6
Prerequisites 6
Installation procedure 6
Samples 6
Building from Source 6
Prerequisites 6
Preparing the build 6
Performing build 7
Switching the Database 7
Installing JDBC-Driver 7
Driver Configuration 7
Changing the dialect 7
Officially supported database engines 7
Switching to MS SQL Server 8
Final notes 8
Playing sample processes 10
Running processes 11
Overtime work 11
Vacation 18
Business trip 20
Authentication 22
Authentication configuration 22
NTLM authentication 22
Active Directory authentication 22
Security system description 22
Definitions 23
Object types 23
Default permissions 24
Configure example: How to create configuration 26
AdminKit 36
Bot Invoker 36
Description 36
Command 36
Configuration 36
Script Runner 37
Description 37
Command 37
Configuration 37
LDAP Importer 38
Description 38
Command 38
Configuration 38
Bots Configuration 39
Introduction to Bot 39
Bot configuration 39
Bot configuration file structure 39
Tag bots 39

Tag task..... 40
Complete bot configuration example..... 40
Task handlers..... 41
DatabaseTaskHandler..... 41
E-mailTaskHandler..... 42
CancelThisProcessInstanceTaskHandler..... 42
MSWordReportTaskHandler..... 42
Running bots..... 43
Appendix A. GNU LESSER GENERAL PUBLIC LICENSE..... 44

Introduction

Runa WFE is an end user Workflow/BPM solution distributed under the terms of LGPL.

This document describes the most common administrative operation with Runa WFE: installation, security configuration, bots configuration, switching the database engine etc. It also briefly discusses building Runa WFE from source and playing sample processes.

Installation

Runa WFE comes as a binary package bundled with a compatible version of JBOSS Application Server. It is also possible to build Runa WFE from source.

Binary distribution

Prerequisites

- JDK 5.0 or higher, can be downloaded from http://java.sun.com/j2se/1.5.0/download.jsp

Installation procedure

0. Download and install JDK 5/0 or higher and set JAVA_HOME environmental variable (http://www.jboss.org/wiki/Wiki.jsp?page=JBossInstallation).
1. Unpack `runa-wfe-*.zip` archive

Runa WFE is ready to go. To start the system, please execute `run.bat` (Windows) or `run.sh` (Unix) from wfe-x.x.x/bin folder.

Navigate your browser to http://localhost:8080/wfe. Default credentials are:

Login: Administrator

Password: wf

Samples

Runa WFE 2.1 comes configured to the embedded hsqldb-database with a bundle of deployed sample processes.

Building from Source

Prerequisites

- JDK 5.0 or higher, can be downloaded from http://java.sun.com/j2se/1.5.0/download.jsp
- Apache Ant 1.6.x, can be downloaded from http://ant.apache.org/bindownload.cgi
- System must be deployed in JBoss AS 3.2.x or Jboss AS 4.0.x which can be downloaded from http://sourceforge.net/project/showfiles.php?group_id=22866&package_id=16942

All other libraries and frameworks are bundled with Runa WFE source distribution and can be found in `lib` subdirectories of the subprojects.

Preparing the build

- Install J2SE SDK and set `JAVA_HOME` environmental variable (http://www.jboss.org/wiki/Wiki.jsp?page=JBossInstallation)
- Install Apache Ant 1.6.x and make `junit.jar` library available for Ant (http://ant.apache.org/manual/OptionalTasks/junit.html)
- Install JBoss Application Server (http://docs.jboss.org/jbossas/admindevel326/html/ch01.html)
  Note: JBoss configuration used in this manual is called `default`
- *Either* using any compatible SVN-client download the source code from the repository *or* download and unpack a source snapshot called runa-wfe-*.*.*-src.zip

- Edit `/build.properties` in the project root folder:

Property `jboss.home.dir` must point to your JBoss installation directory
for Windows:

```
jboss.home = C:/jboss-4.0.x
```

for Unix:

```
jboss.home = /opt/jboss-4.0.x
```

Performing build

Run ANT installation script in the root folder of the project

```
ant install.wfe
```

This command will build and install Runa WFE into your JBoss installation folder.

Switching the Database

By default Runa WFE is configured to use the embedded hsqldb database. Jboss.org/wiki says that “hsqldb is not a production quality database. It is suitable for demos and testing. JBoss ships with the database to help you get something working out of the box”. So does Runa WFE. If you plan to use Runa WFE for production environment you should consider using another database engine for persistence.

Let's discuss the procedure of database switching.

Installing JDBC-Driver

First of all you need a database JDBC-driver. JDBC-driver is usually a jar-file. There exists a variety of JDBC-drivers and it is possible to find a suitable one for virtually any database engine.

Some vendors ship their databases with a JDBC-driver. Sometimes it is necessary to use third-party driver.

Once you've found a suitable driver put it under `${jboss.home}/server/default/lib` folder. This makes the driver accessible for Runa WFE.

Driver Configuration

Once the driver is installed the datasource for Runa WFE should be configured to use this driver and the driver itself should be provided a connection URL and some other configuration parameters. In order to achieve this `${jboss.home}/server/default/deploy/runawfe-ds.xml` should be edited.

The exact contents of the file depends on the selected JDBC-driver. A number of useful examples covering a vast majority of popular databases is available in `${jboss.home}/docs/examples/jca` folder of the original jboss distribution.

Changing the dialect

After all you need to change the hibernate dialect used to talk to the database. Edit `${jboss.home}/server/default/conf/hibernate.cfg.xml` and set the dialect property:

```
<property
name="dialect">net.sf.hibernate.dialect.ADialectForYourDatabase</property>
```

Runa WFE is ready to go.

Officially supported database engines

Currently Runa WFE supports hsqldb, MS SQL Server 2000/2005 and MySQL. Oracle 10 support is planned in the upcoming 2.1 release.

Switching to MS SQL Server

In the case of MS SQL Server the most suitable driver is jTDS. It can be found in a `wfe/lib` folder of the source distribution. As discussed previously the driver should be put in `${jboss.home}/server/default/lib` folder.

In the case of MS SQL Server `${jboss.home}/server/default/deploy/runawfe-ds.xml` should look something like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
  <local-tx-datasource>
    <jndi-name>runawfe-ds</jndi-name>
    <connection-url>jdbc:jtds:sqlserver://<server>[:<port>][/<database>][;<pr
operty>=<value>[;...]]</connection-url>
    <driver-class>net.sourceforge.jtds.jdbc.Driver</driver-class>
    <user-name>sa</user-name>
    <password>**</password>
      <metadata>
         <type-mapping>MS SQLSERVER2000</type-mapping>
      </metadata>
  </local-tx-datasource>
</datasources>
```

In the simplest case the `connection-url` property looks like this:

```
<connection-url>jdbc:jtds:sqlserver://localhost/runawfe</connection-url>
```

Please refer to the jTDS documentation for details on settings other properties if required.

As a hibernate dialect `net.sf.hibernate.dialect.SQLServerDialect` is used:

```
<property name="dialect">net.sf.hibernate.dialect.SQLServerDialect</property>
```

Switching to MySQL

MySQL Connector/J is the official JDBC-driver for MySQL. As usually the driver should be put in `${jboss.home}/server/default/lib` folder.

`${jboss.home}/server/default/deploy/runawfe-ds.xml` should look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
  <local-tx-datasource>
    <jndi-name>runawfe-ds</jndi-name>
    <connection-url>jdbc:mysql://<server>[:port]/<database>?useUnicode=true&a
mp;characterEncoding=UTF-8</connection-url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <user-name>user</user-name>
    <password>****</password>
  </local-tx-datasource>
</datasources>
```

Hibernate's dialect for MySQL is `net.sf.hibernate.dialect.MySQLDialect`.

Final notes

You may want to try Runa WFE with different database engines. The general procedure is already described:

- Find a suitable JDBC-driver and put it in the `lib` folder;

- Edit the `runawfe-ds` datasource configuration;
- Change the hibernate dialect used to talk to the database.

In a Perfect World this is just enough to run RunaWFE over a new database engine. But in a real life some problems may (and possibly would) appear. Please don't hesitate to ask questions in our forum and also share your experience with other people.

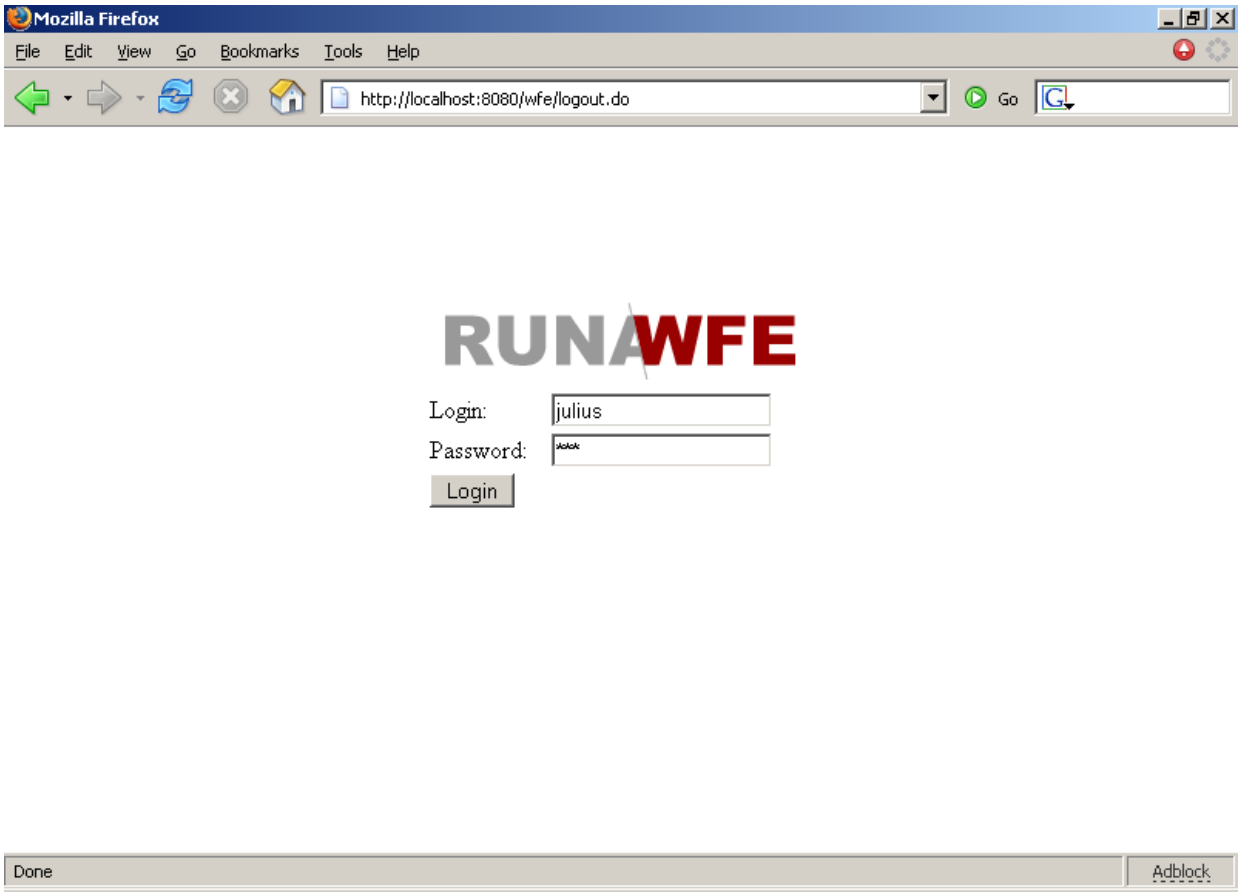
If you want to integrate a new database environment into automated build process (for example to run tests over different from hsqldb database) you have to modify `runawfe-ds.xml` and `hibernate_build.properties` files of the source distribution (please, don't commit them). You also require to make sure your JDBC-driver is always located at right place. The simplest way to do this is to put it under the `wfe/lib` folder of the source distribution.

Playing sample processes

- Surf to http://localhost:8080/wfe
- Accounts:

Login	Full name
julius	Gaius Julius Caesar
nero	Nero Claudius Caesar
cleopatra	Cleopatra VII The Daughter of the Pharaoh
caligula	Gaius Iulius Germanicus
tiberius	Tiberius Claudius Drusus
marcus	Marcus Aurelius Antoninus
gaiua	Gaiua Flavius Valerius Constantinus
attila	Attila the King of Huns

The password for sample accounts is 123



sample groups:

Group	Group members
manager	julius nero

human resource	cleopatra
bookkeeper	caligula tiberius
staff	marcus gaiua attila
all	all sample actors

Menu items:

- System – allows to:
  - Manage system permissions
- Process Definitions – allows to:
  - Start business process
  - View permissions on business process definition
  - View the business process graph
- Executors – allows to:
  - Manage executor permissions
  - Manage executor properties
- Process Instances – allows to:
  - View process instance states and variables
- Tasks – allows to:
  - Work with actors tasks

Sample processes:

- Overtime work
- Vacation
- Business trip

Notes:

- Only managers (*julius*, *nero*) have the rights to start processes "Overtime work" and "Business trip".
- All sample actors can start the process "vacation".
- The boss[1] of *gaiua* is *julius*, the boss of *marcus* and *attila* is *nero*.

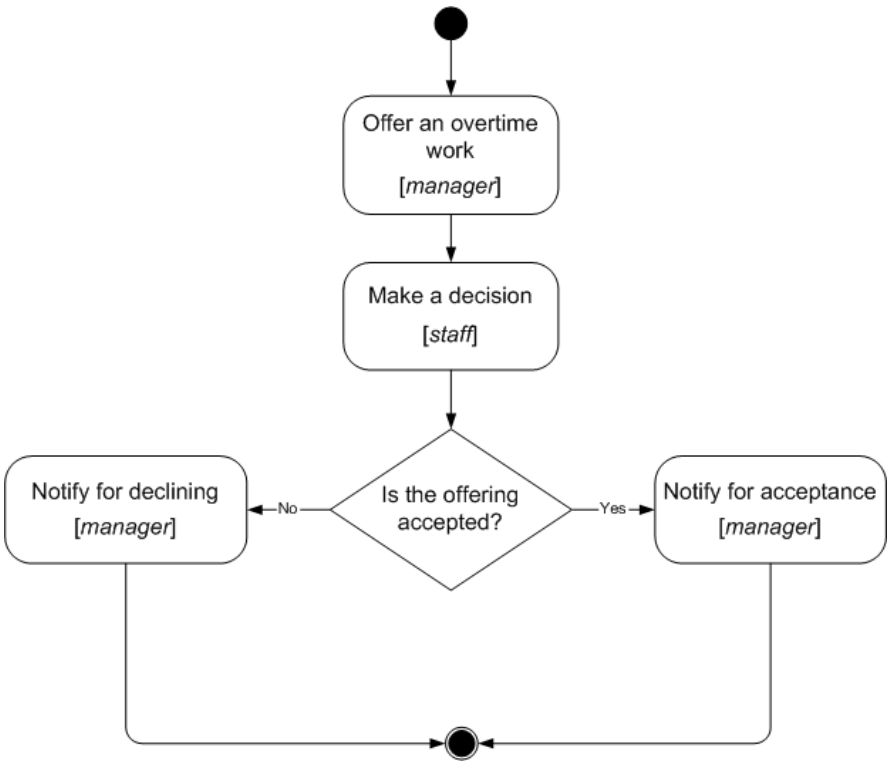
Running processes

Overtime work

Description:
Manager asks the employee for overtime work. The employee accepts or declines the proposal. After that the manager receives the notification of acceptance or rejection.

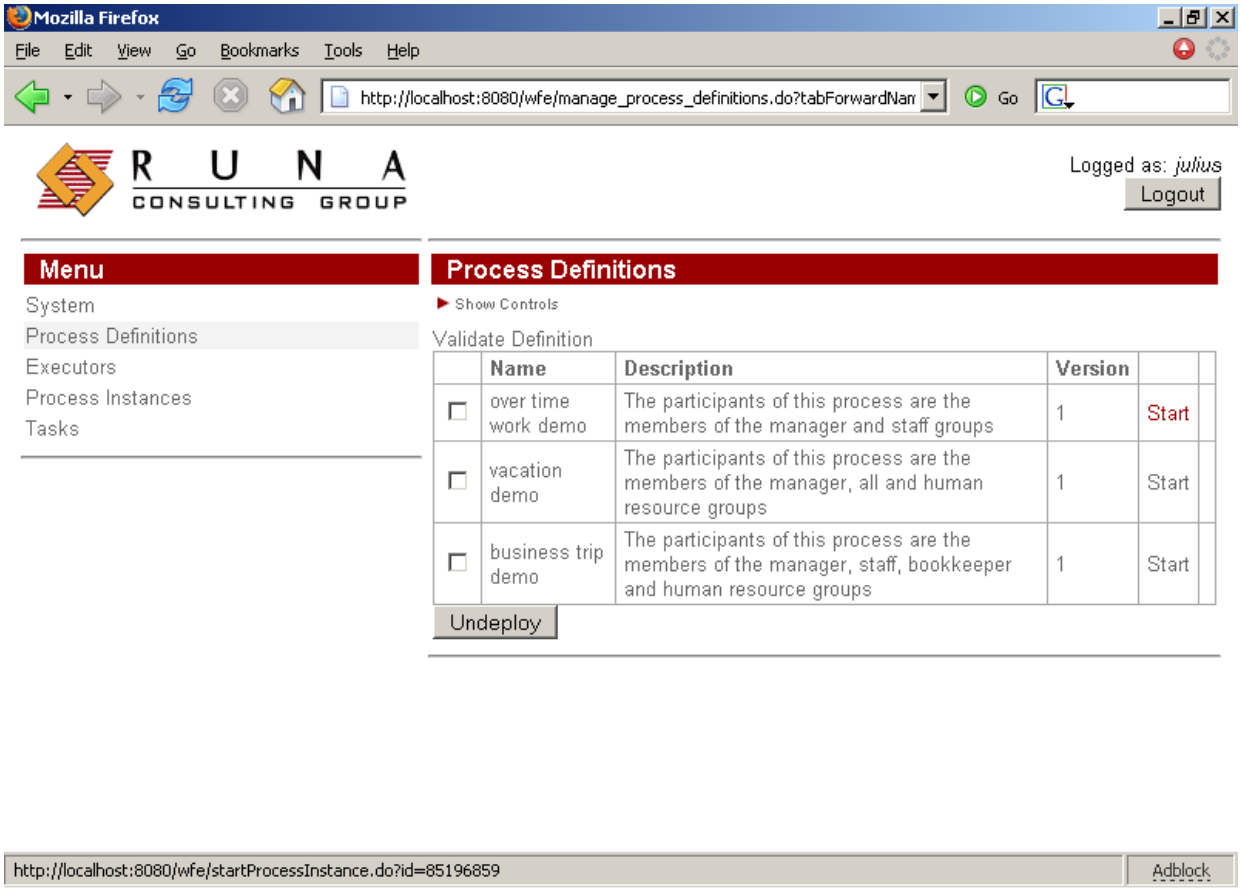
The business process graph:

1 Organization function determines boss in "vacation" process.



How to play the business process:

Log in as member of manager group (e.g. julius)
Click menu item “Process Definitions”
Click “Start” for “overtime work”

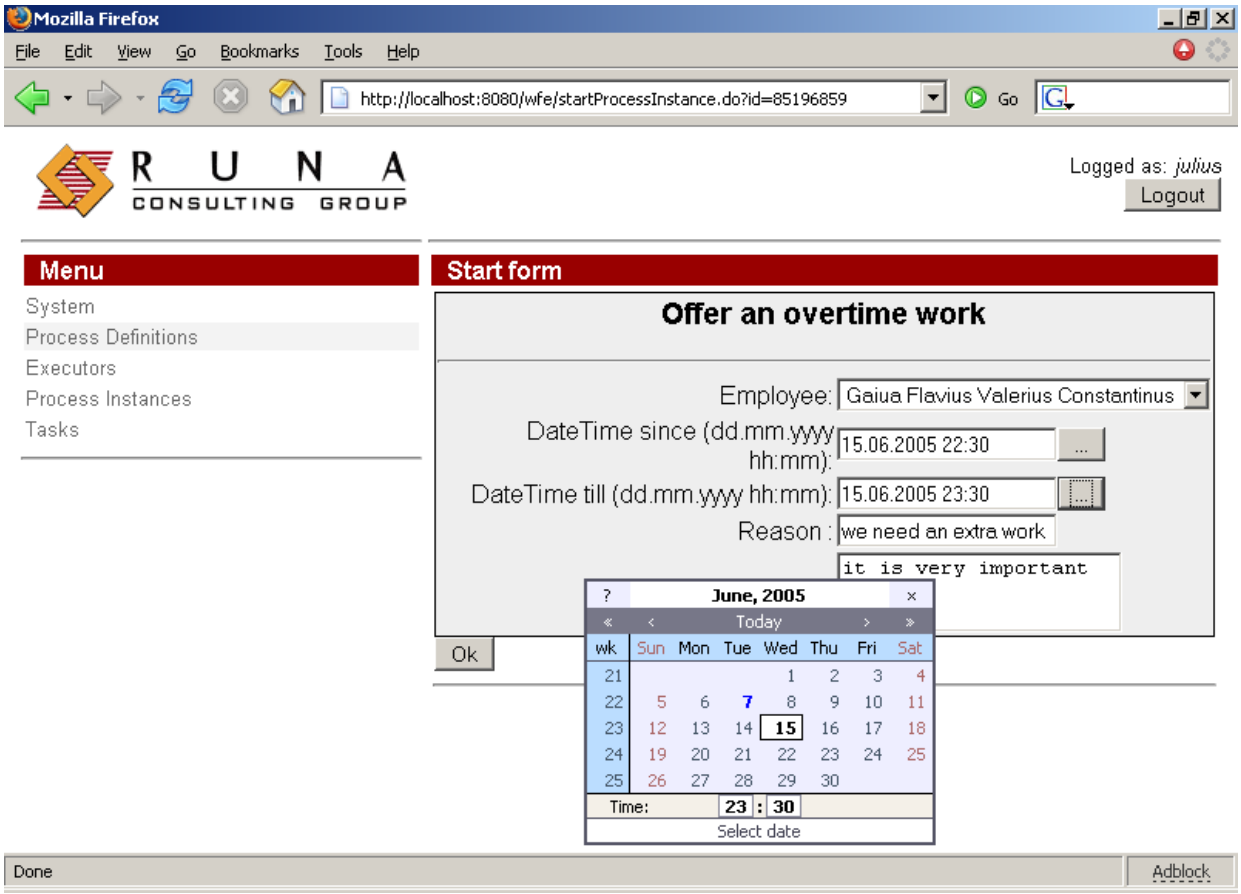


Start form will appear.

- Choose an employee, which will receive proposal for an overtime work

(e.g. gaiua)

- Enter the time interval for the overtime work (e.g. 15.06.2005 22:30 - 15.06.2005 23:30)
- Fill the fields «reason» and «comment»
- Click “OK”

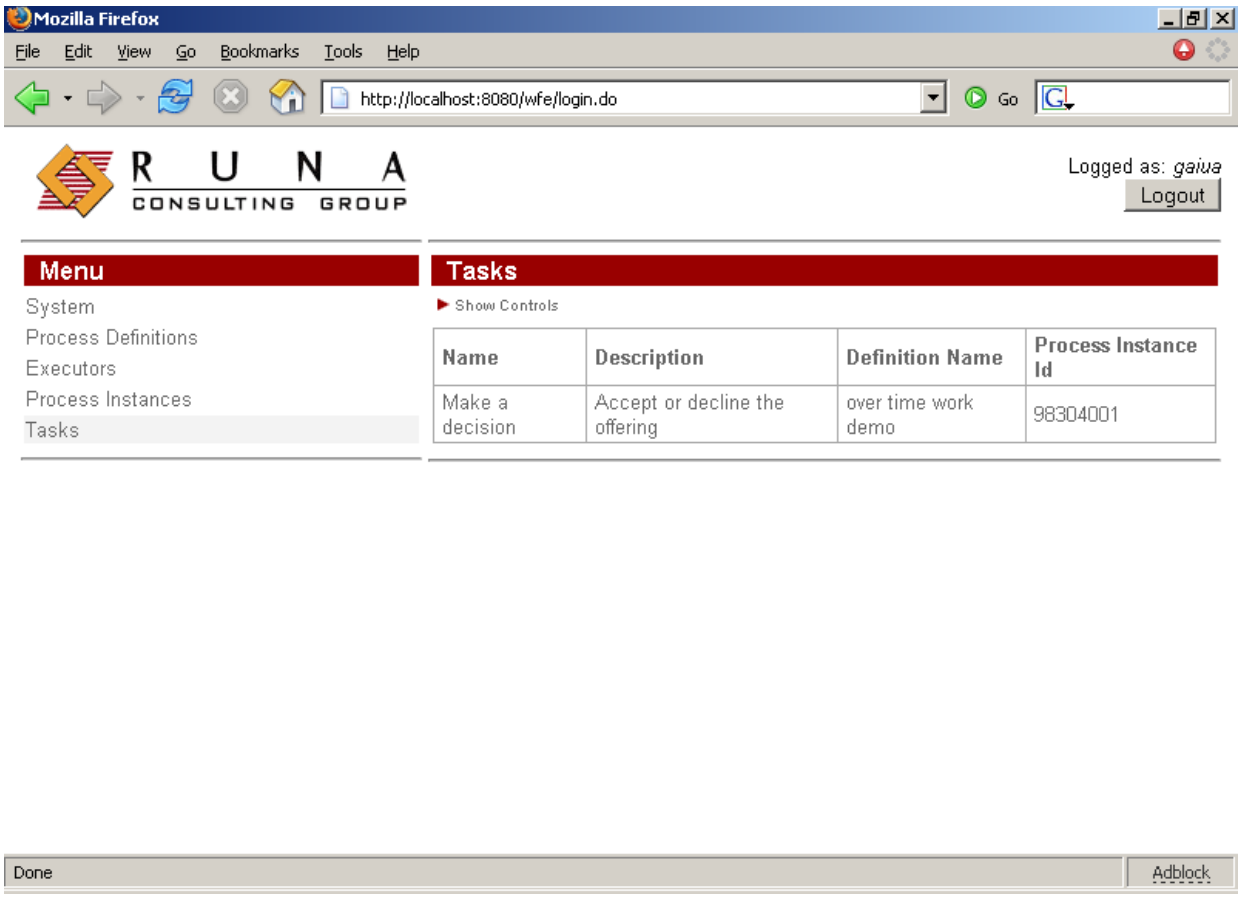


The new instance of “overtime work ” business process will be created and the employee (gaiua) will receive task.

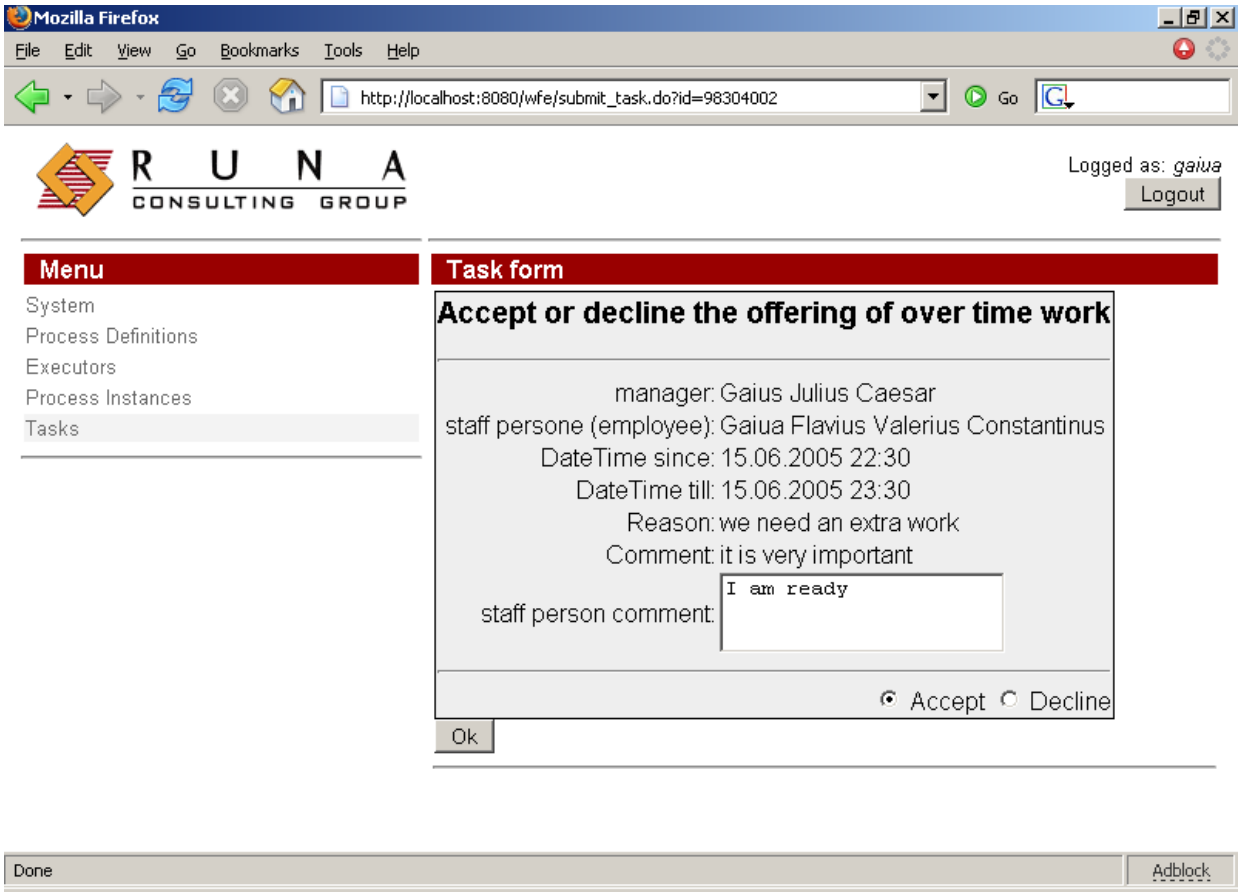
Click “Logout”

Log in as employee, selected on the start form (gaiua).

Click the “make a decision” task.

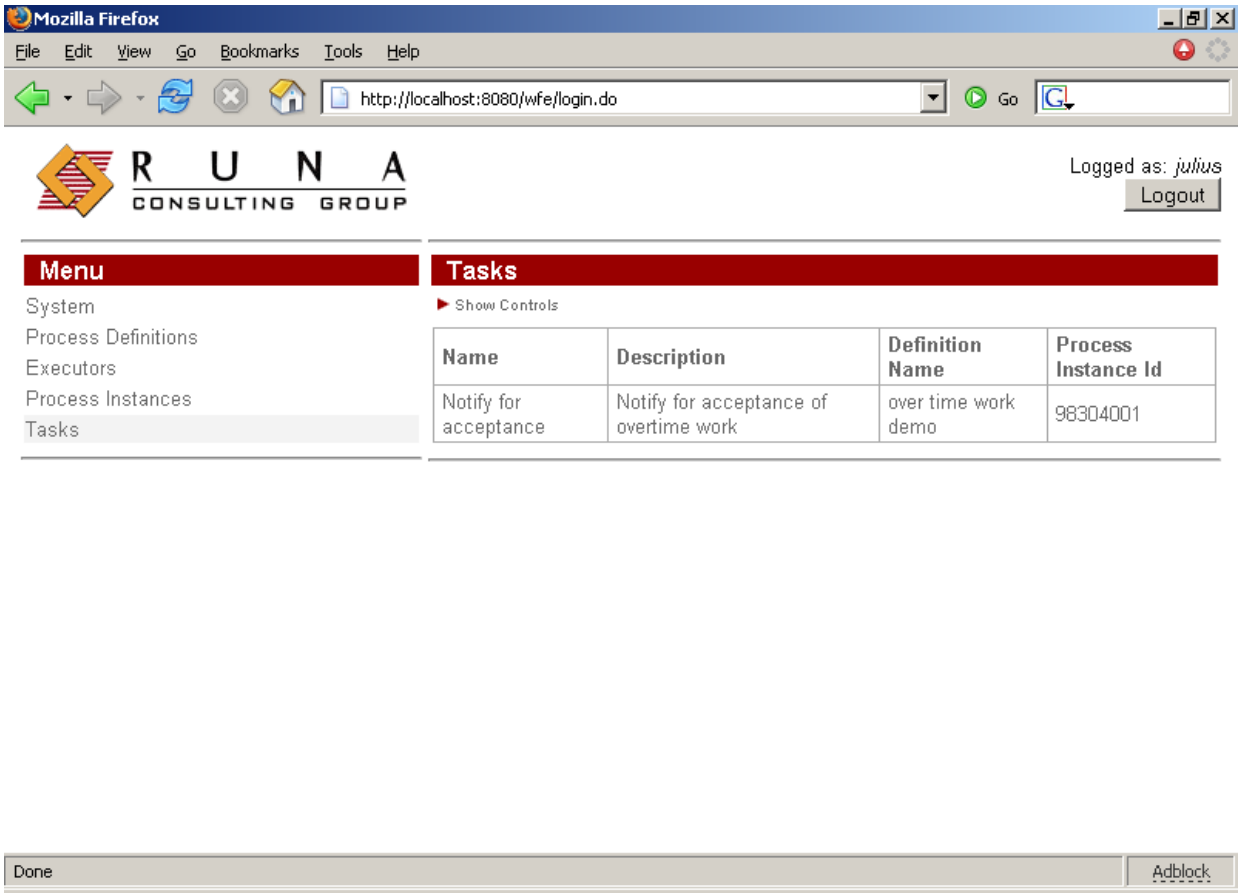


Choose “accept” or “decline” (e.g. “accept”), fill the «comment» field and click «OK».

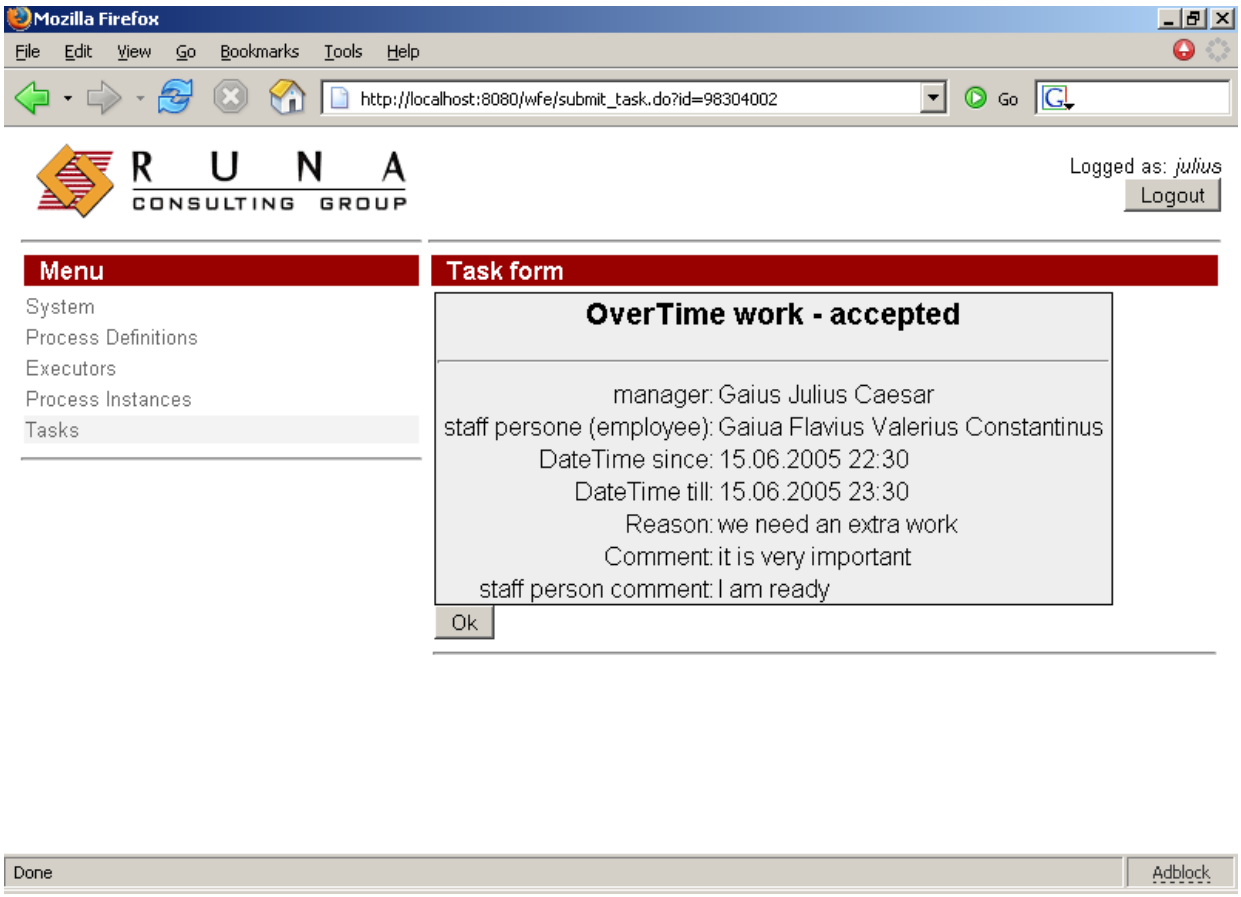


Logout

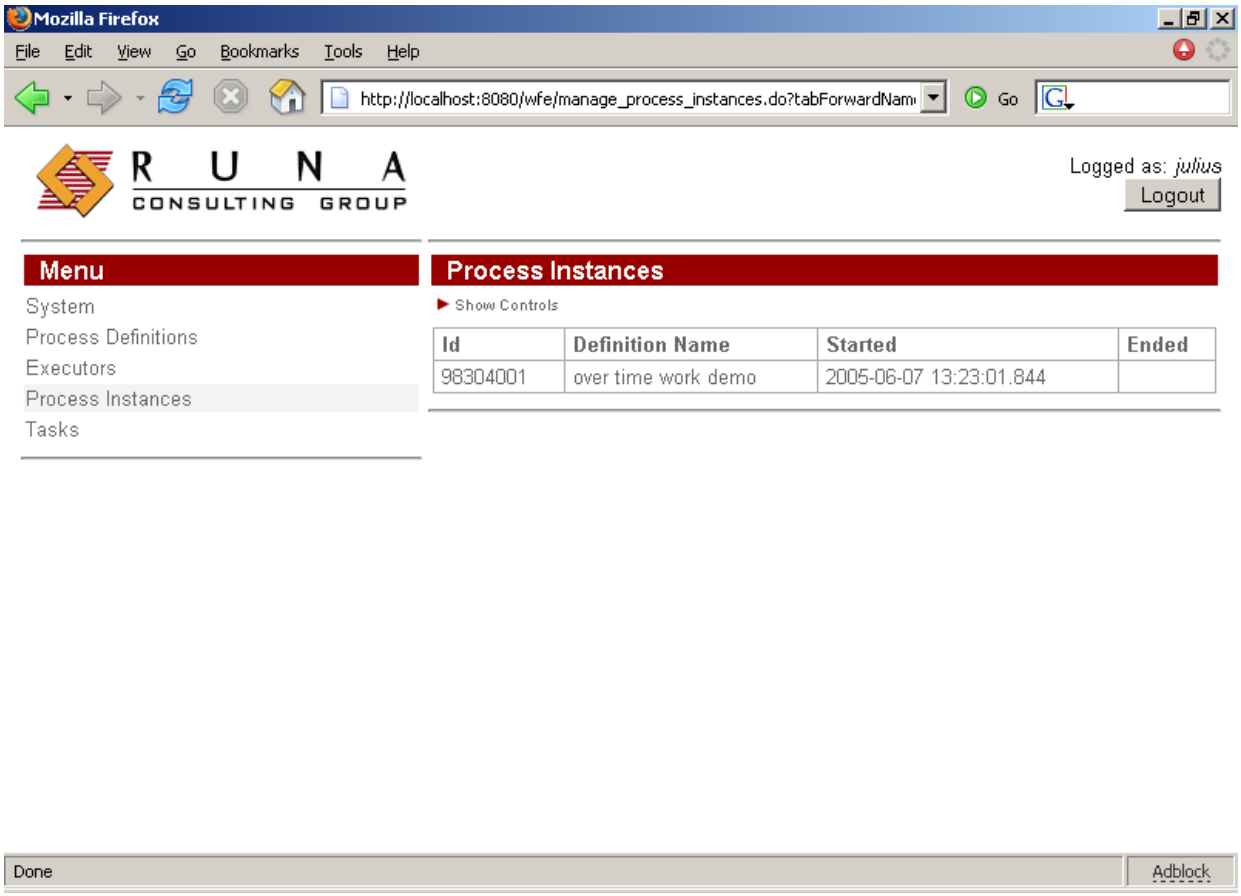
Log in as an actor, which started the process (julius)
Click on task “Notify for acceptance”.

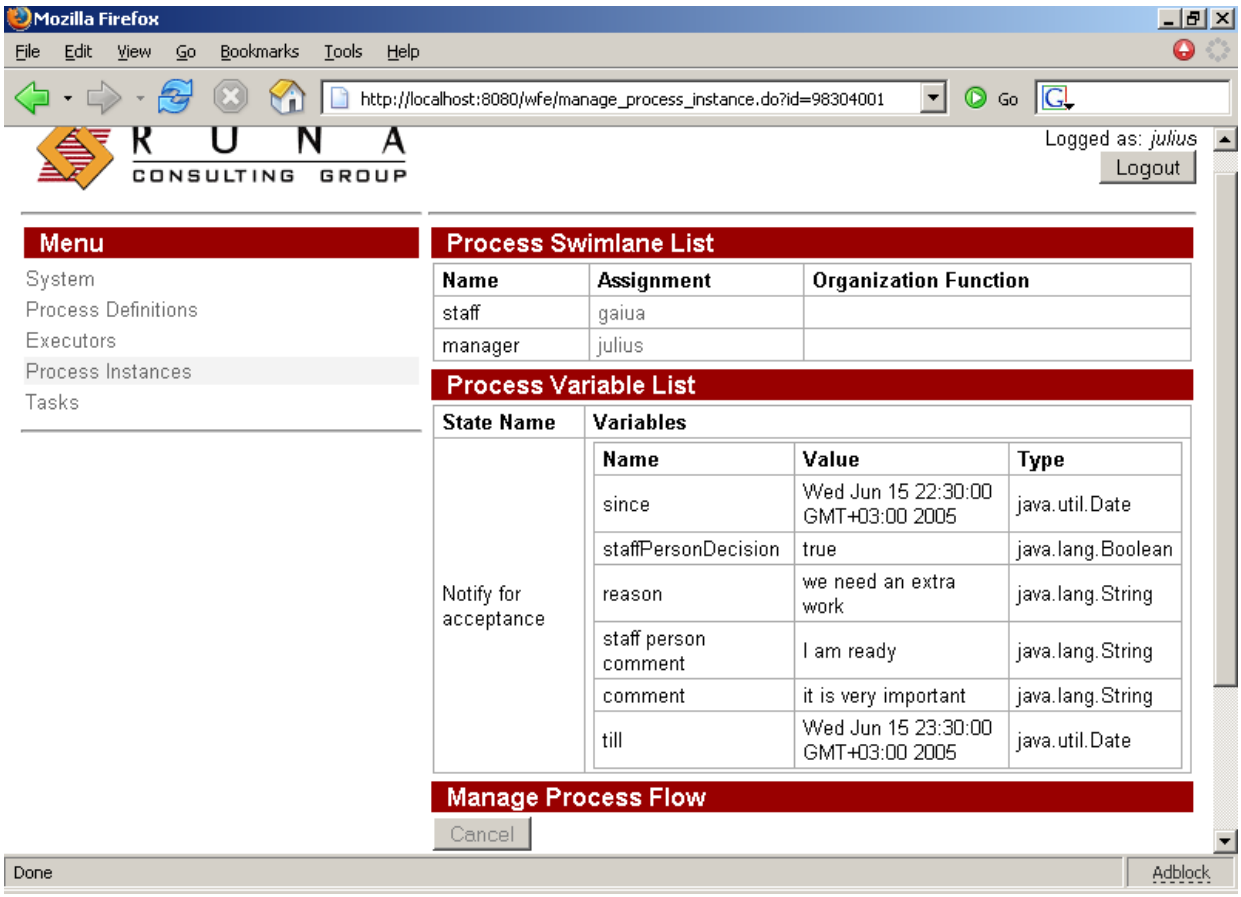


By clicking “OK” you’ll finish the process.

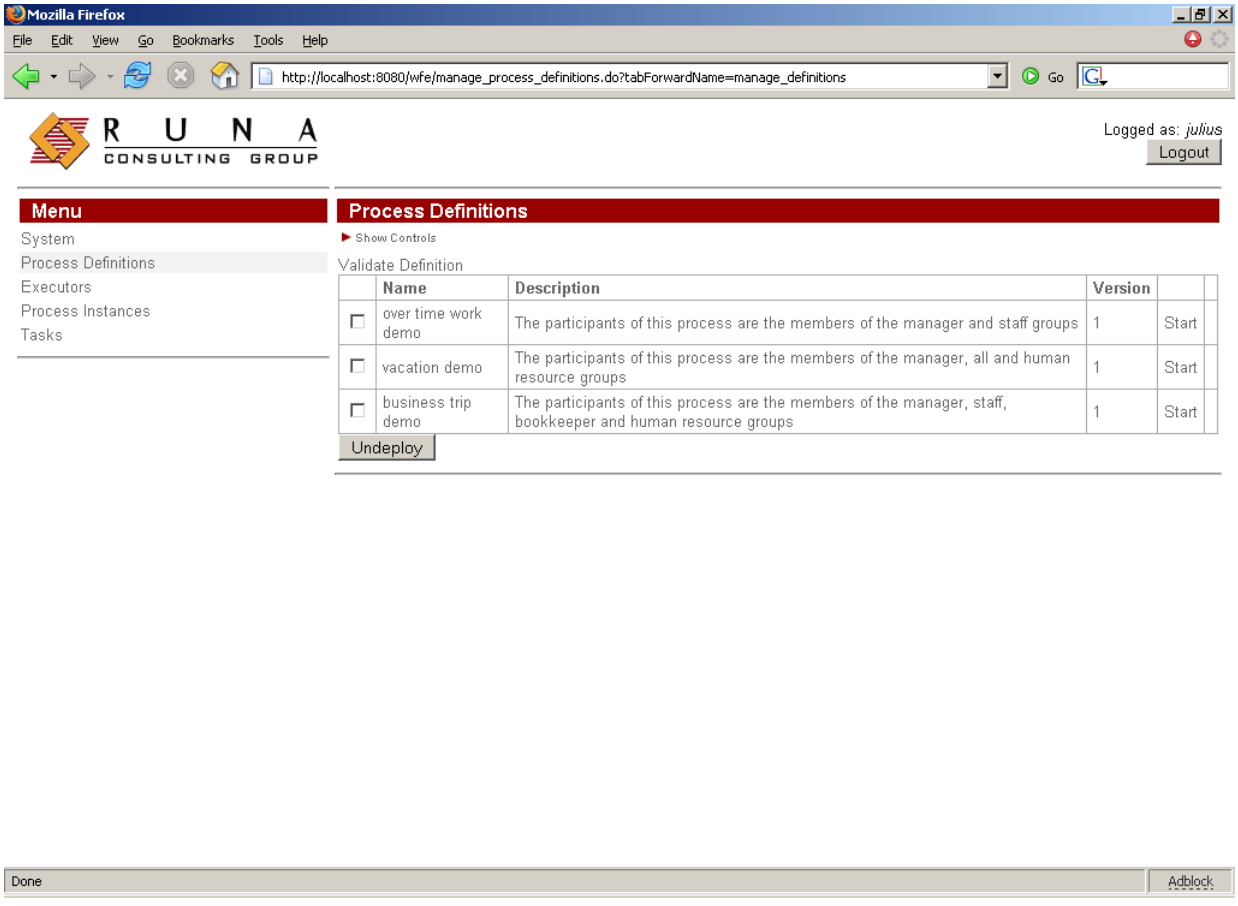


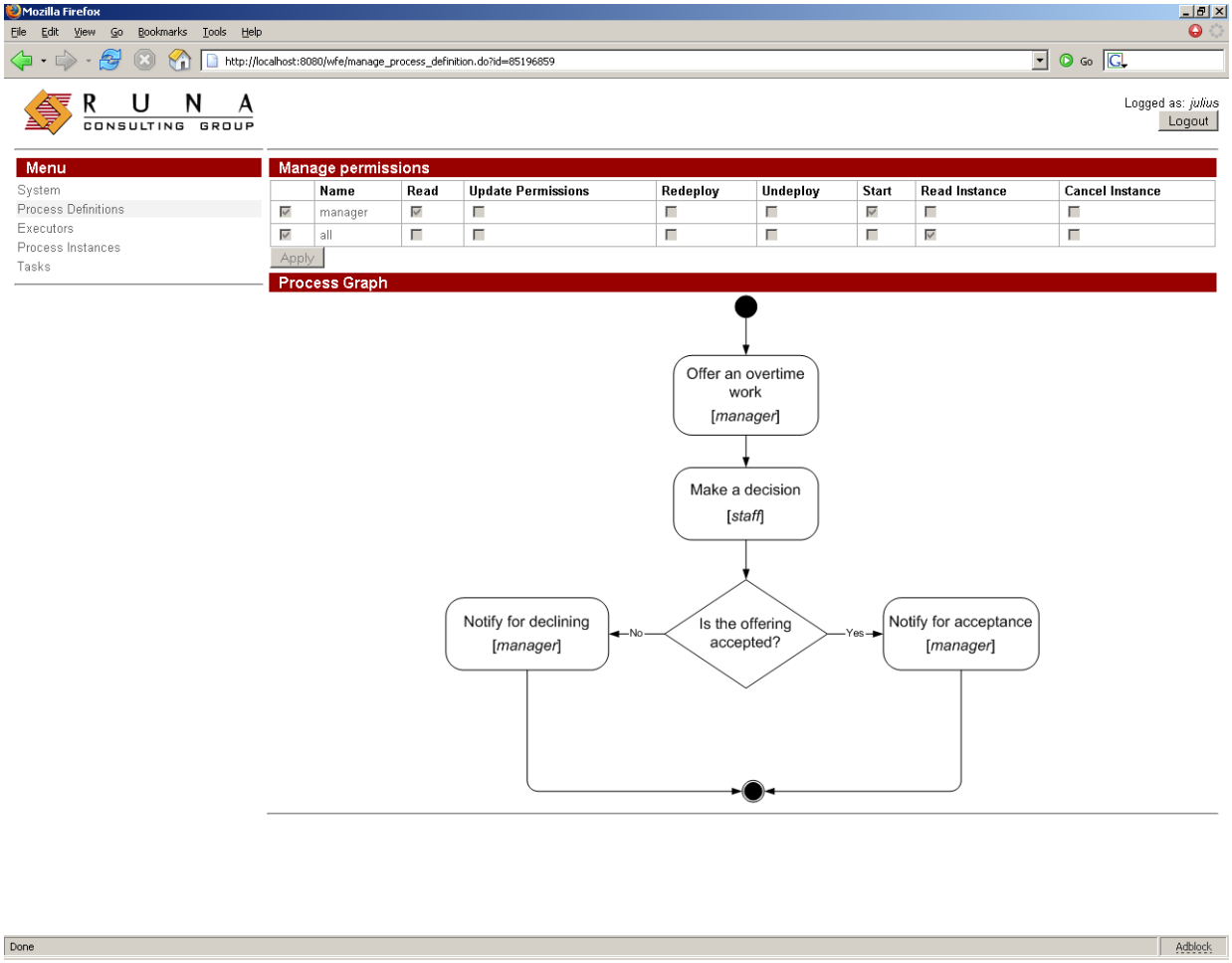
Note. To monitor process click on “Process Instances” menu item and click on instance.





Note. To see the business process graph click on “Process Definitions” menu item and click on process.





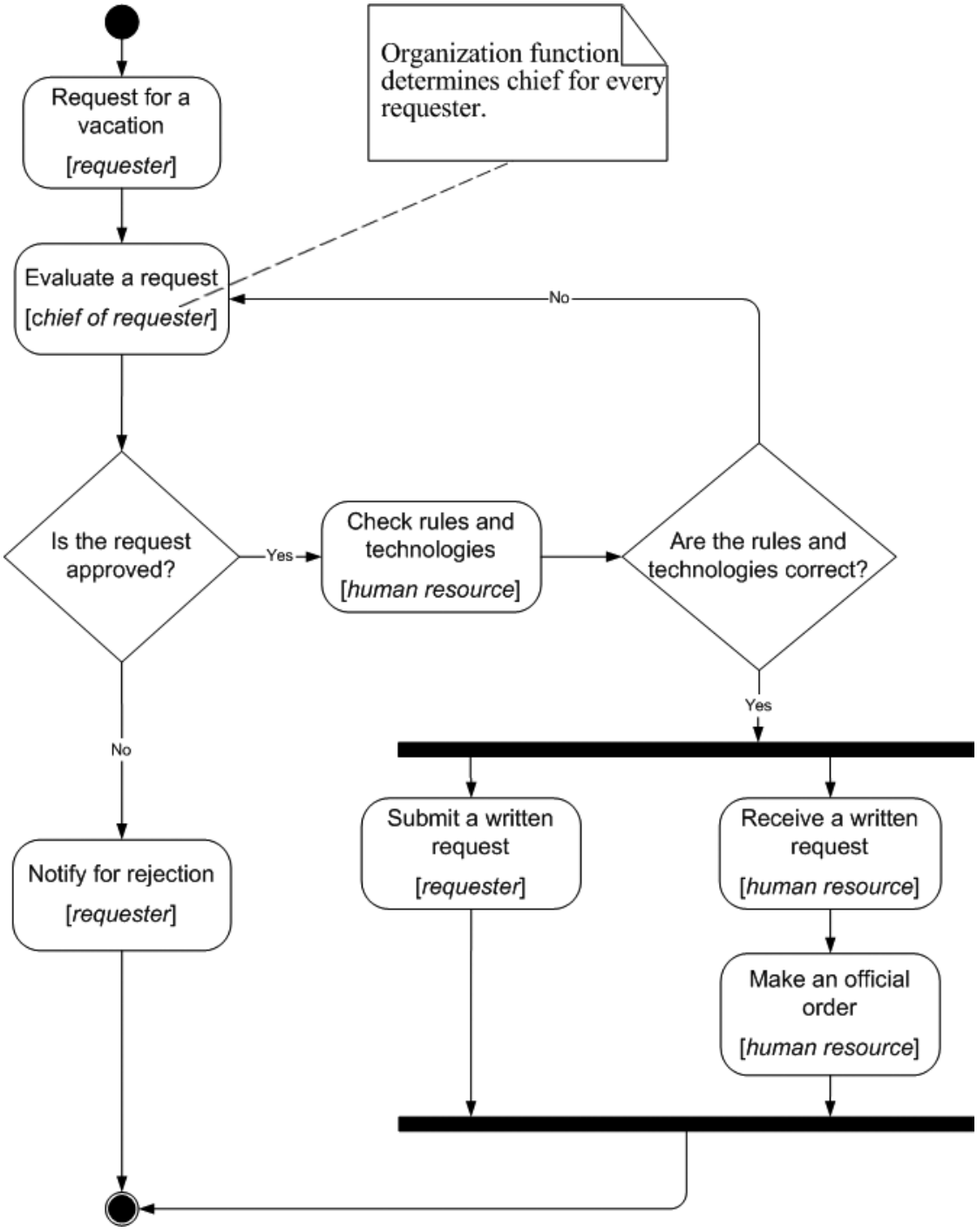
Vacation

Description:

An employee requests for vacation. The boss[2] accepts or declines the request. If request was rejected, the employee receives notification and business process ends. If request was approved, the human resource inspectors receive task “check rules and technologies”. Human resource inspector checks business process data. If check result is negative, the business process returns to the “evaluate a request” state (the cycle example). Boss accepts or declines the request. If the check result is positive, the business process sends task for hardcopy request submission and official order issuing. Employee receives task “submit the written request for a vacation” to the human resource department”, the human resource inspector receives task “receive a written request from employee” concurrently. Upon completion this task the human resource inspector receives next task “make an official order”. After all these tasks are done, the business process ends.

2Organization function determines boss for every actor. Actors with login names starting with “g” have boss *julius*, the others – *nero*.

The business process graph:



How to play the business process:

Log in as member of *staff* group (e.g. *marcus*)
Click on the menu item "Process Definitions"
Click "Start" for "vacation"
Start form will appeared. In this form:

- Enter the time interval for the vacation
- Fill the fields «reason» and «comment»
- Click "OK"

The new instance of "vacation" business process will start and boss receives the "evaluate a request" task. The boss of *marcus* is *nero*

Click "Logout"

Login as *nero*
Click on “evaluate a request” task
Choose “accept”, fill the «boss comment» and click «OK»
Click “Logout”
Login as member of the “human resource” group. (*cleopatra*)
Click “check rules and technologies” task
Choose “correct”, fill the “human resource inspector comment” and click “OK”
Click on “Receive a written request” task
Click “OK”
Login as *marcus*. Click on “Submit a written request” task and then click “OK”
Login as *cleopatra*. Click on “Make an official order” task.
The form will appeared. In this form:

- Fill the “Official order number” field
- Fill the “Official order date” field
- Click “OK”

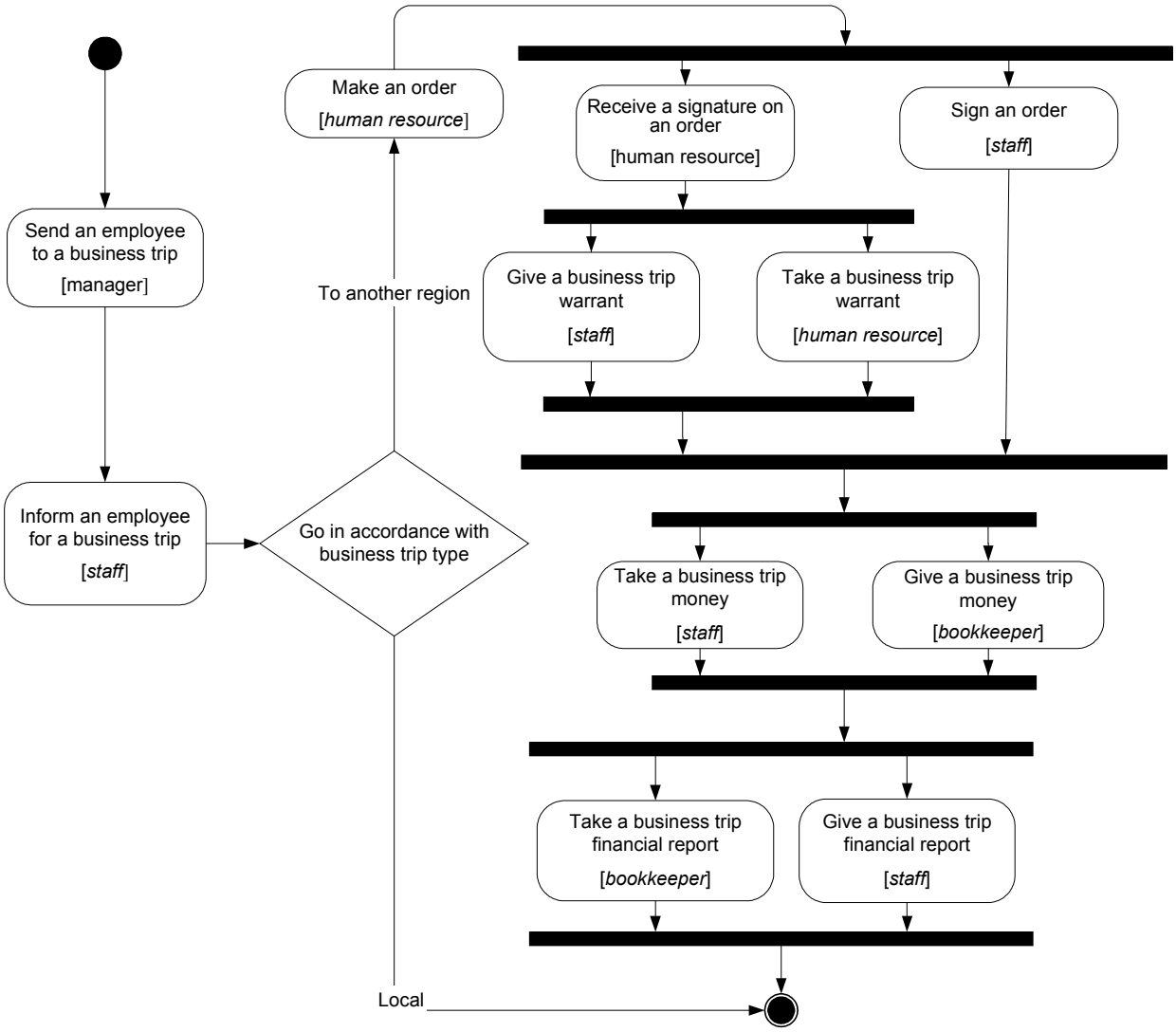
The business process ends

Business trip

Note. This process is quite complex. This documentation contains only short description and the business process graph.

The short description:
A manager sends an employee to a business trip. The employee receives notification. If the business trip is local, the business process ends. If business trip type is “to another region”, a human resource inspector receives task “make an official order”. The employee receives task “sign an order” and human resource inspector receives a task “receive a signature on the order” concurrently. Employee receives task “receive a business trip warrant” and inspector receives a task “give a business trip warrant”. Bookkeeper issues and employee receives the business trip money. After employee returns from the business trip, employee submits and bookkeeper receives the business trip financial report. The business process ends.

The business process graph:


Send an employee to a business trip
[manager]
Inform an employee for a business trip
[staff]
Go in accordance with business trip type
To another region
Make an order
[human resource]
Local
Receive a signature on an order
[human resource]
Sign an order
[staff]
Give a business trip warrant
[staff]
Take a business trip warrant
[human resource]
Take a business trip money
[staff]
Give a business trip money
[bookkeeper]
Take a business trip financial report
[bookkeeper]
Give a business trip financial report
[staff]

Authentication

Authentication configuration

Runa WFE uses JAAS for authentication. Main configuration file for authentication is `login_module.properties`. This file is plain .properties file which defines login modules and their requirements.
Runa WFE provides following modules:

- `ru.runa.af.authenticaion.InternalDBPasswordLoginModule` – module authenticates username and password against internal Runa WFE database
- `ru.runa.af.authenticaion.ADPasswordLoginModule` – module authenticates username and password against Microsoft Active Directory server
- `ru.runa.af.authenticaion.NTLMLoginModule` – module authenticates NTLM authentication digest against Windows domain PDC

NTLM authentication

NTLM login module uses two configuration files: `ntlm_support.properties` and `ad_password_login_module.properties`. These files are located in `<server name>/conf` directory.
`ntlm_support.properties` options:

- `domain` – domain name which PDC is used for authentication, only required if NTLM support is enabled
- `ntlm_supported` – enable or disable NTLM support (enabled if true, disabled otherwise)

`ad_password_login_module.properties` options:

- `ru.runa.af.active.directory.damain.name` – domain name which PDC is used for authentication, only required if NTLM support is enabled. Must be the same as in `ntlm_support.properties`.

`ntlm_support.properties` file required only for WEB client, and not required if thick client is used.
With enabled NTLM, users can authenticate via `http://<servername>:<port>/wfe/ntlmlogin.do` page.

Active Directory authentication

Active Directory login module uses single configuration file – `ad_password_login_module.properties`. This file can be found in `<server name>/conf` directory.
Configuration options:

- `ru.runa.af.active.directory.damain.name` – domain name which users module tries to authenticate
- `ru.runa.af.active.directory.server.url` – URL of Active Directory server

Security system description

Definitions

System – the workflow system
Executor – actor or group of actors, the performers that can perform actions with system
Own executor permissions– executor permissions that granted to executor itself
Inherited executor permissions– sum of permissions of all executor groups
Executor permissions– sum of own and inherited executor permissions

Object types

The main concept of Runa WFE security system is a secured object. Secured object is an object that can have security permissions applied to it. There are several type of predefined secured objects:

- System
- Executors
- Business process definitions
- Business process instances

Each type of secured objects has unique set of permissions which can be applied to it.

Secured object type	Permissions
System	Read
	Update Permissions
	Login
	Create Executor
	Deploy Process Definition
Executor	Read
	Update Permissions
	Update Executor
Group[3]	List
	Add to Group
	Remove from Group
Process definition	Read
	Update Permissions
	Undeploy
	Redeploy
	Start Process Instance

3 Group inherits all permissions from executor.

Secured object type	Permissions
	Read Process Instance
	Cancel Process Instance
Process instances	Read
	Update Permissions
	Cancel

Only specified set of permissions can be applied to secured object of a given type. Application of another set of permissions is prohibited.

Default permissions

There is number of predefined groups that are always present in the system. These groups (privileged executors) have special treatment by Runa WFE system. Upon creation of new secured object privileged executors are granted all available permissions for this secured object type.

Privileged executor	Secured object type
Group *Administrators*	System
	Executor
Group *Process Definition Administrators*	Process Definition
	Process Instance

Privileged executors cannot be deleted and permissions can't be revoked from them. *Administrator*[4] is a member of all privilege groups.

To login into the system an actor must have *Login* permission on *System*.

To create new executor, an actor must have *Create Executor* permission on *System*. Newly created executor has *Read* permission on itself or if executor is group *Read* and *List Group* permissions. Creator is granted all possible permissions on new executor.

Newly created executor has no *Login* permission on object *System*, i.e. can not login to the system. In order to login an actor must have *Login* permission on *System* and have password specified.

An actor must have *Read* permission to read secured object details (executor details, system permissions, process definition details or monitor process instance).

To update permissions on any secured object an actor must have *Update Permissions* on object. An actor also must have *Read* permission on executor which is granted (or revoked) new permission.

An actor with permission *Update* can update executor details or delete executor.

---

4 It is possible to delete *Administrator* but this could lead the system to unstable state

An actor must have *List Group* permission on group to list executors of group.
Only an actor with permission *Add to Group* can add new executors to group (an actor must have *Read* permission on added executors). Only an actor with permission *Remove from Group* can remove executors from group (an actor must have *Read* permission on removed executors).

To deploy new process definition into the system an actor must have *Deploy Process* permission on *System*. Deployer is granted all possible permissions on deployed definition.
Definition can be undeployed by an actor with *Undeploy* permission on it, or redeployed by an actor with *Redeploy* permission.

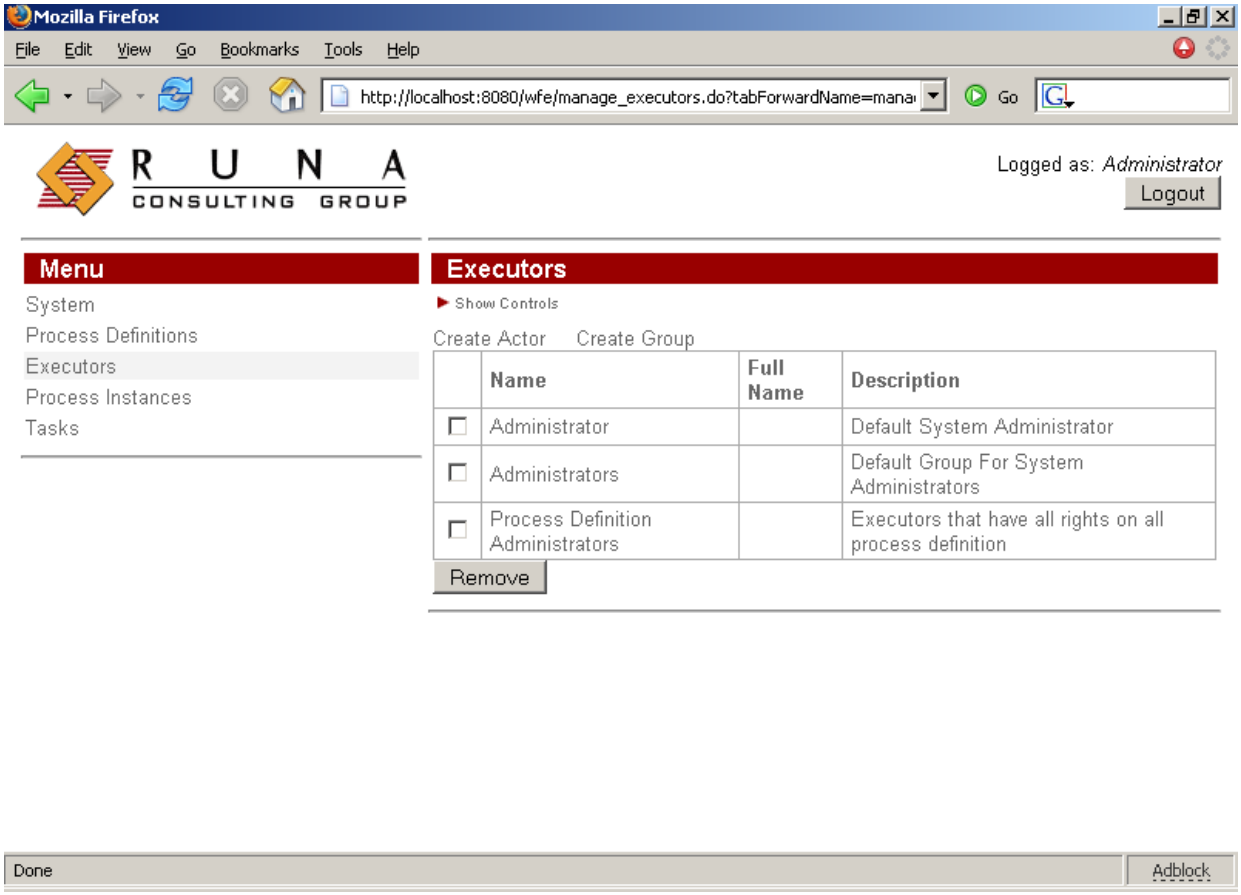
Executors with *Start Process Instance* permission on definition can start new process instance of this definition.

An actor must have *Read* permission on process instance to monitor instance state and variables.
An actor with *Cancel* permission on process instance can cancel process instance execution.

Permissions *Read Process Instance* and *Cancel Process Instance* on process definition determine executors that can start or cancel newly started instance. This applies only on newly started instances after these permissions granting.

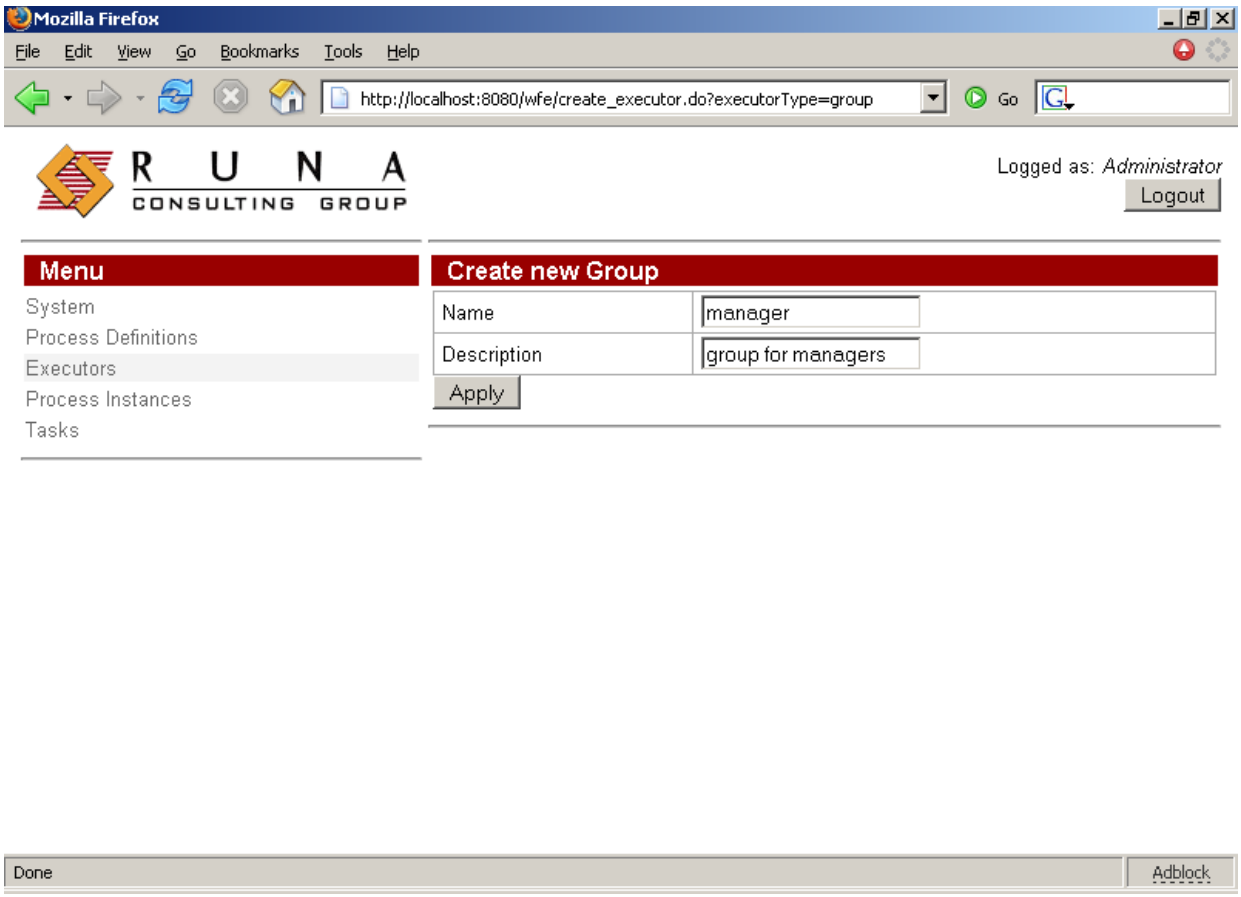
Configure example: How to create configuration.

- Login as Administrator. (The default password is wf)
- Click on the menu item “Executors”



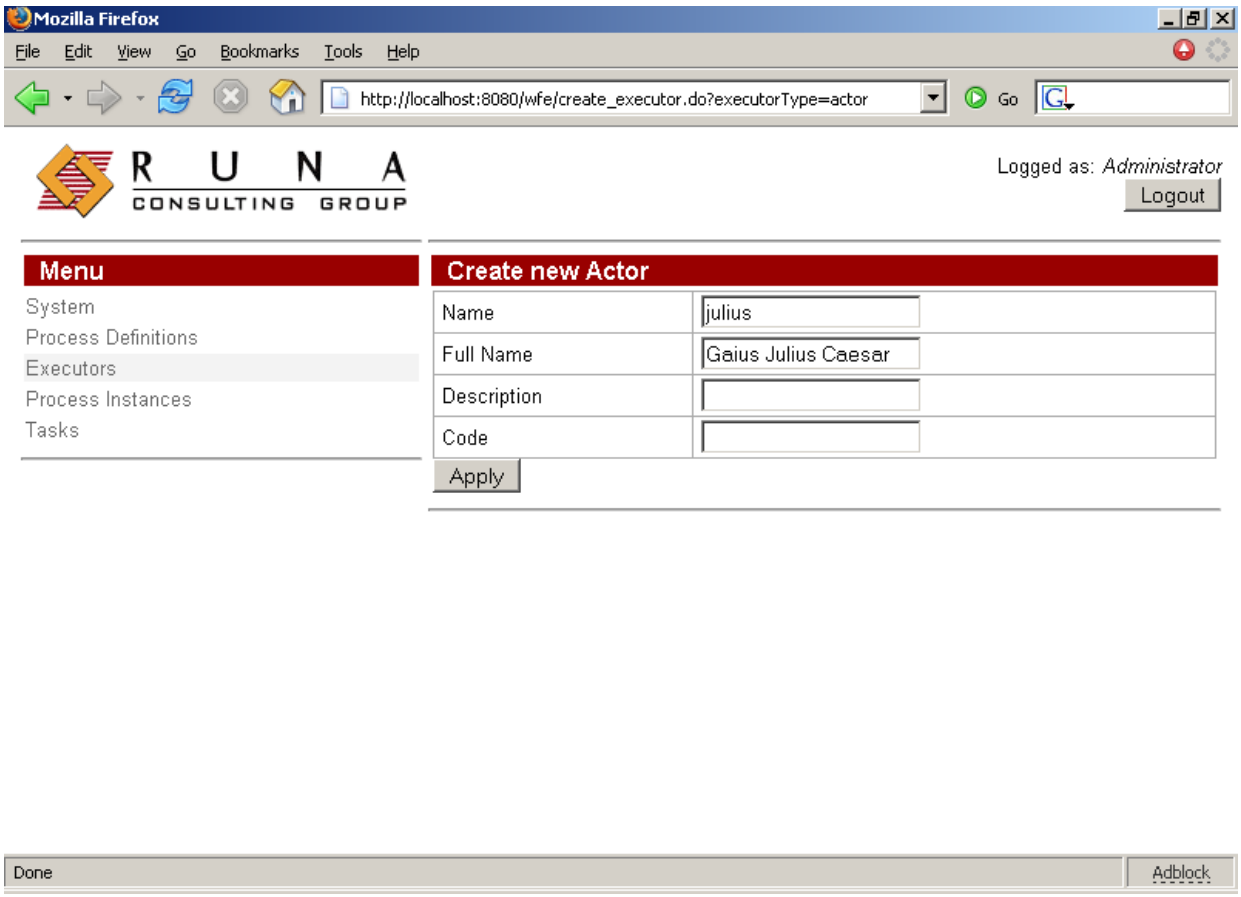
Using “Create Group” command create the following groups:

- manager
- human resource
- bookkeeper
- staff
- all

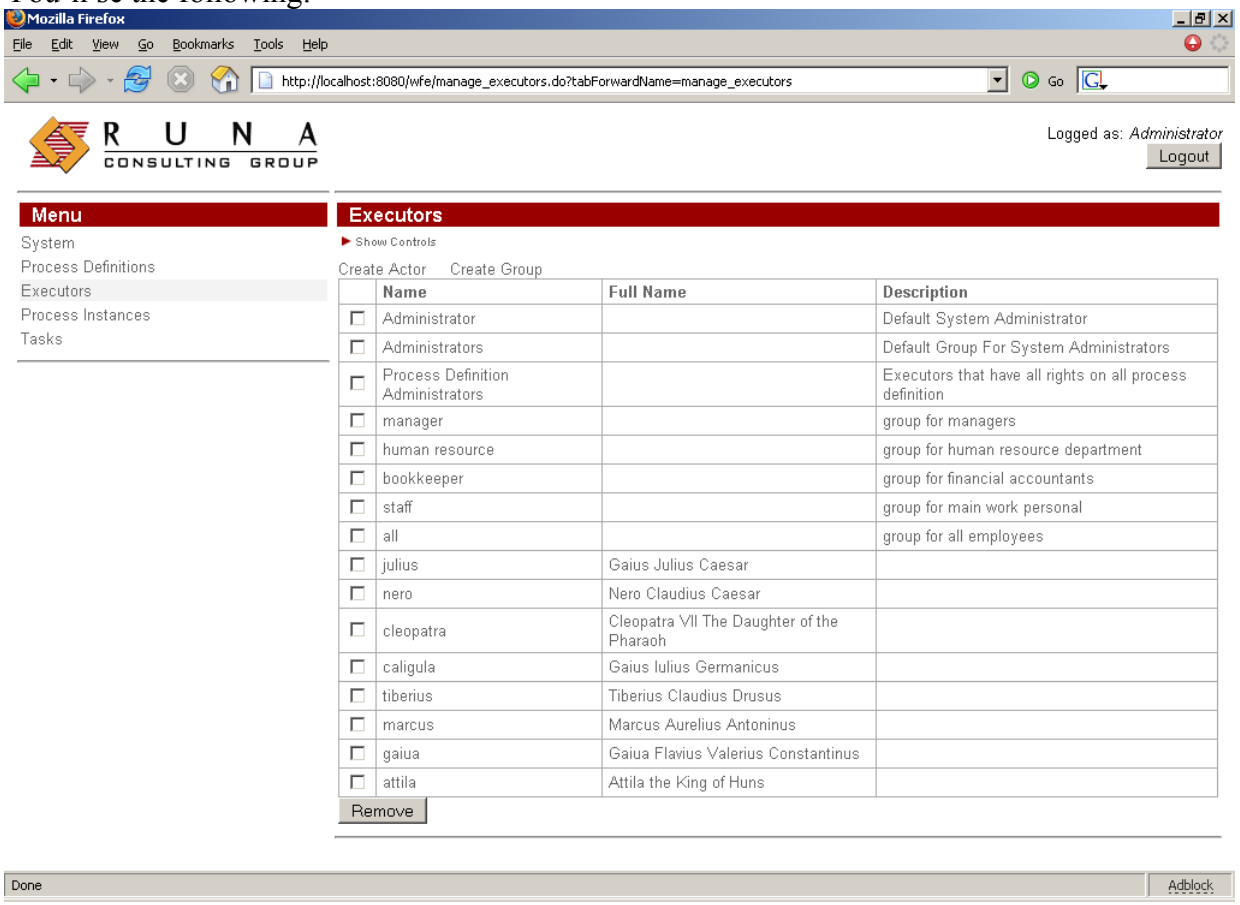


Using “Create Actor” command create the following actors:

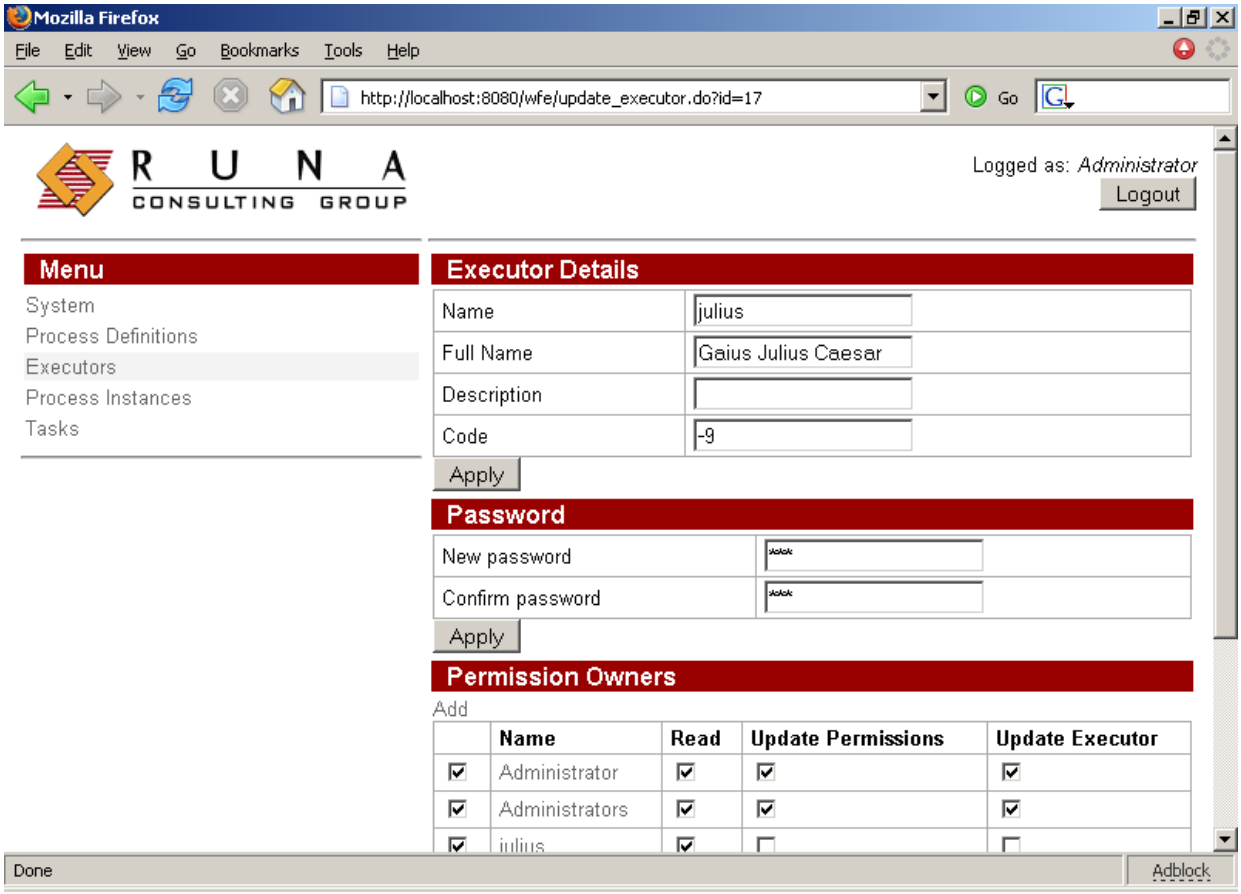
- julius
- nero
- cleopatra
- caligula
- tiberius
- marcus
- gaiua
- attila



You'll se the following:

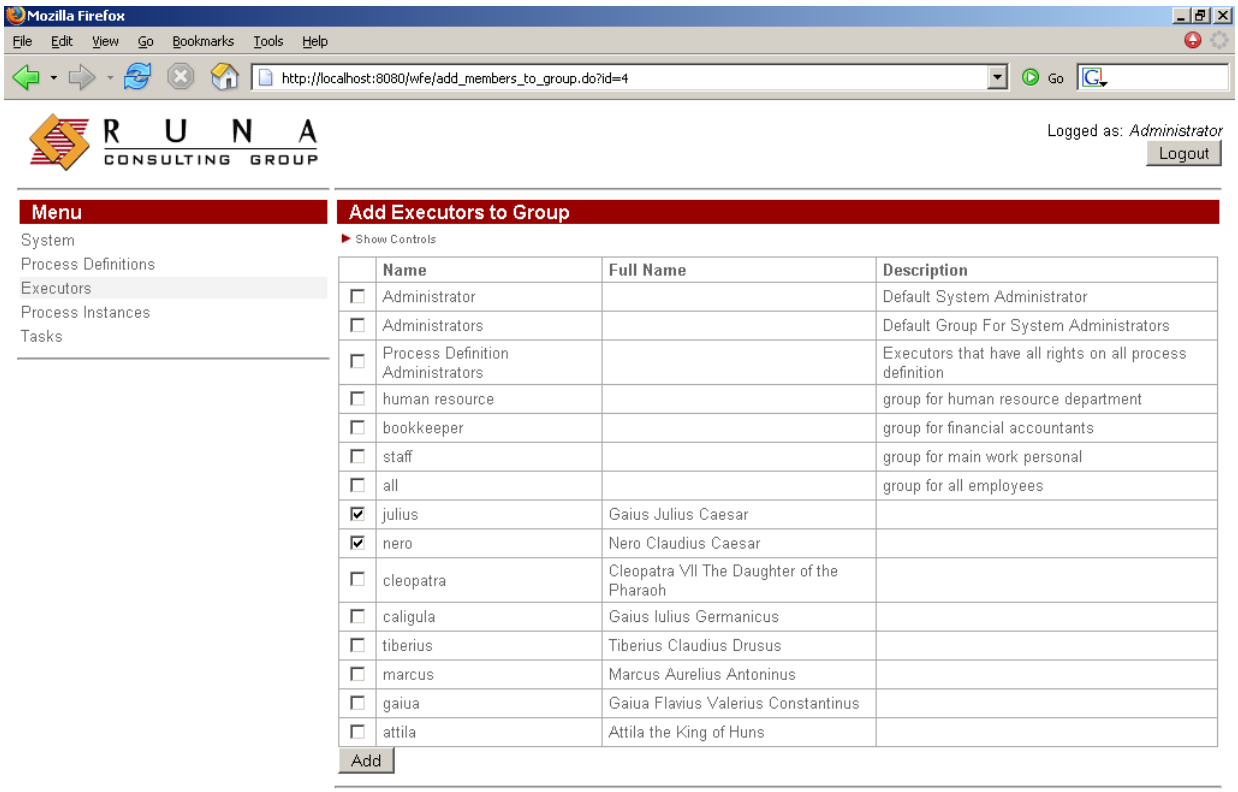


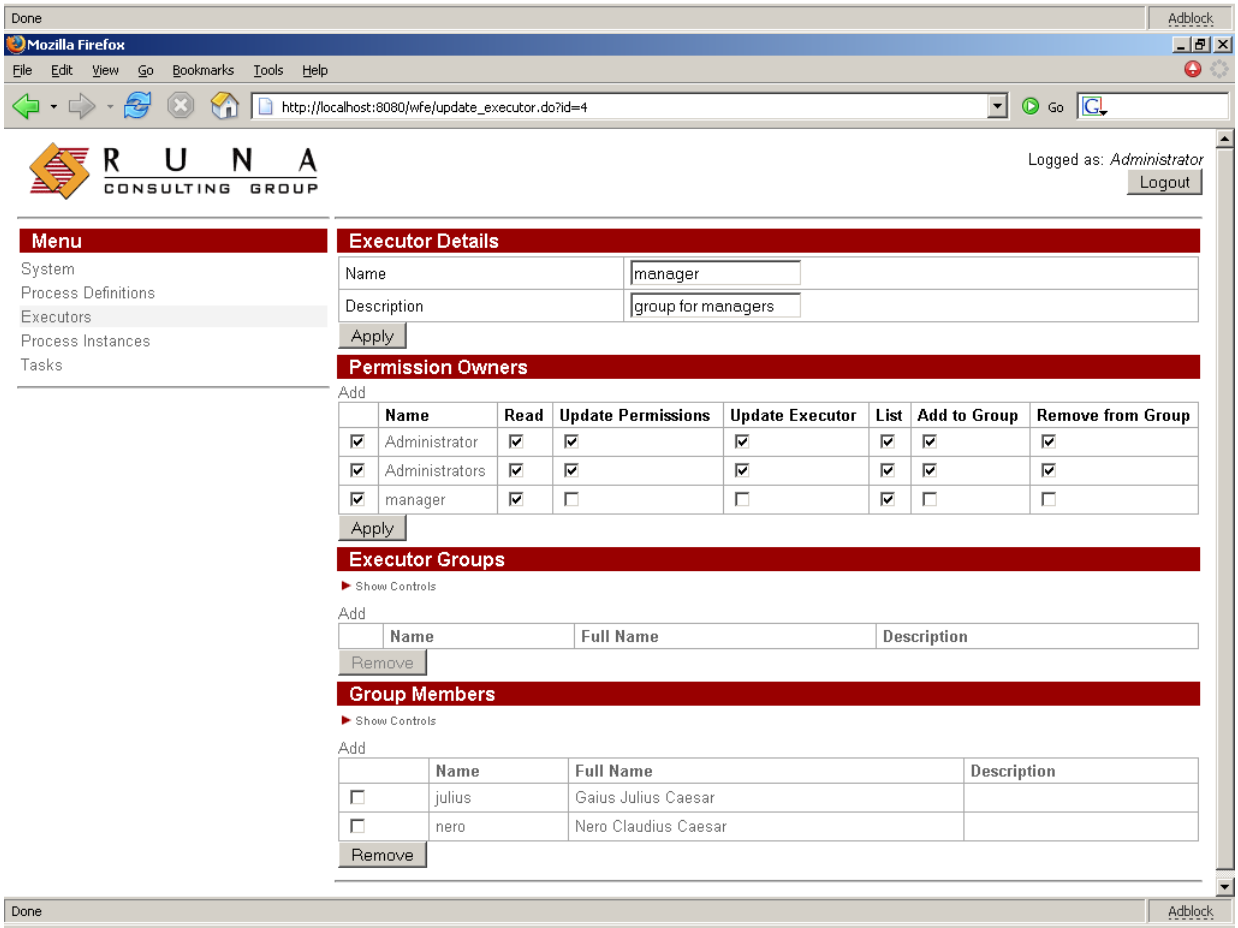
Set the password 123 for all actors



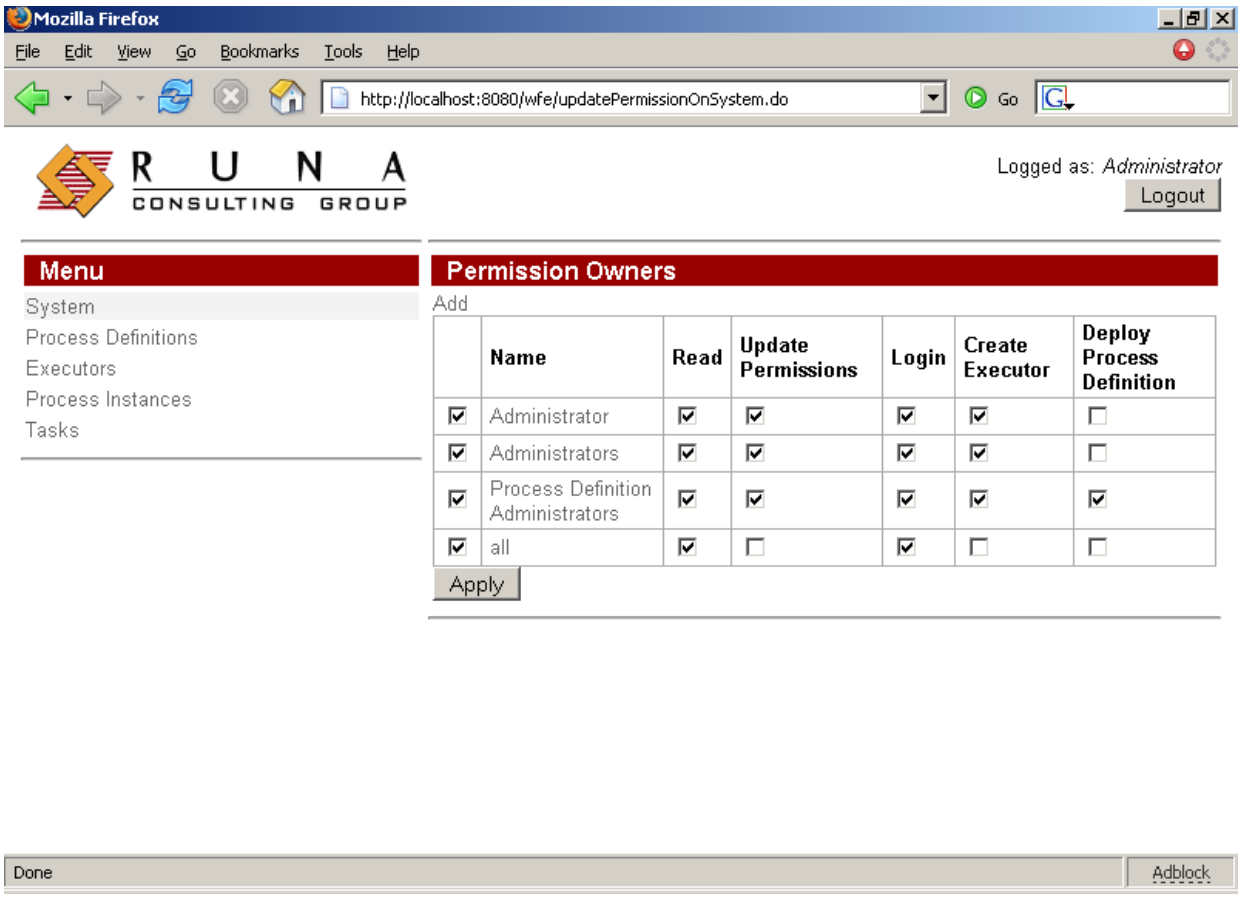
Add actors in groups

Groups	Group members
manager	julius nero
human resource	cleopatra
bookkeeper	caligula tiberius
staff	marcus gaiua attila
all	julius nero cleopatra caligula tiberius marcus gaiua attila

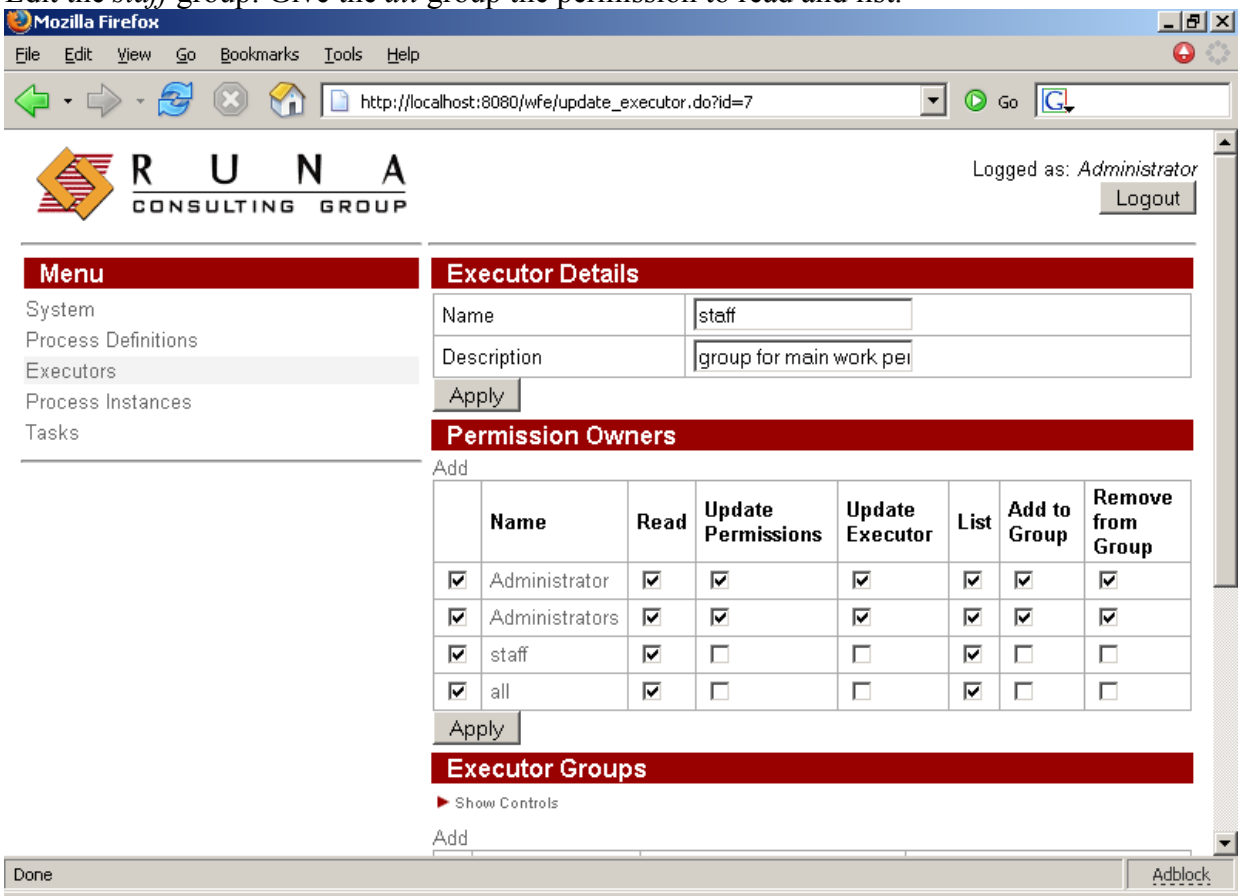




Click the menu item “System”, and give the group *all* login and read permissions.

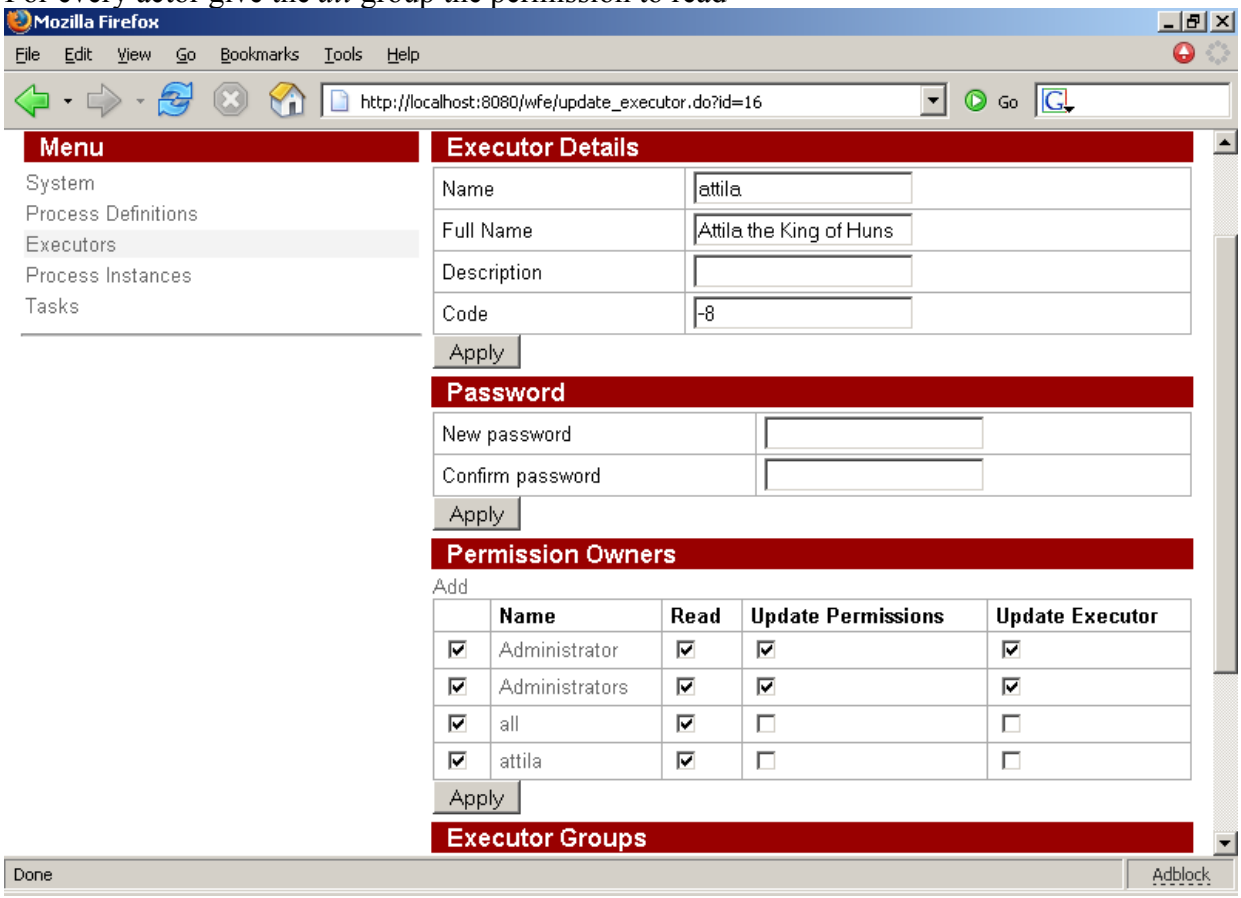


Edit the *staff* group: Give the *all* group the permission to read and list.



Do the same for groups *manager, human resource, bookkeeper*: give the *all* group the

permissions to read and list.
For every actor give the *all* group the permission to read



Click the menu item “Process Definitions”. Execute command Deploy Definition for files

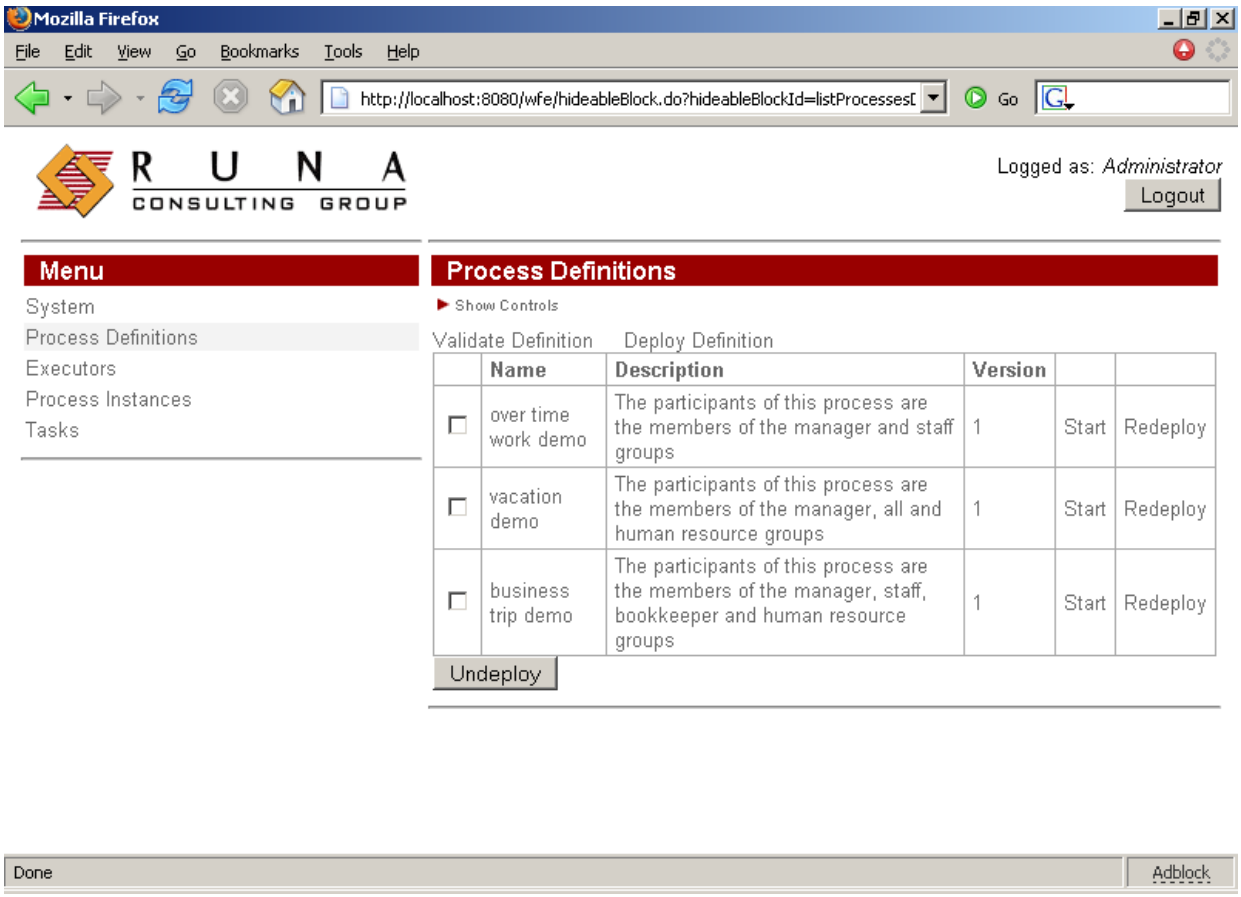
- OverTime.jar
- Vacation.jar
- BusinessTrip.jar

Sample process files can be found in runawfe-x.x.x/samples directory.

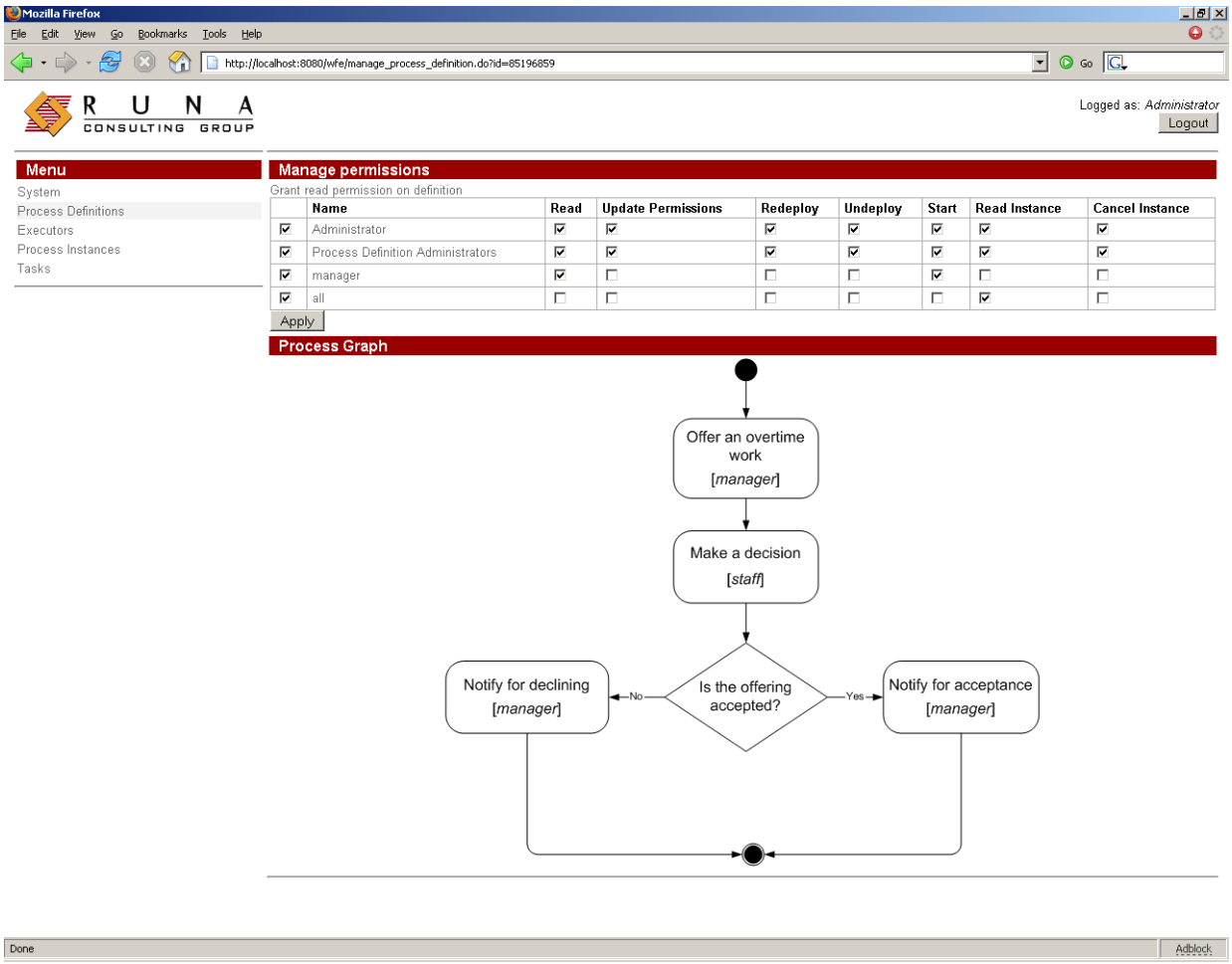
To make process jar go to process directory and run jar[5] cf process.jar .
e.g.

```
cd Vacation
jar cf Vacation.jar .
```

5 Jar utility can be found in java installation folder.

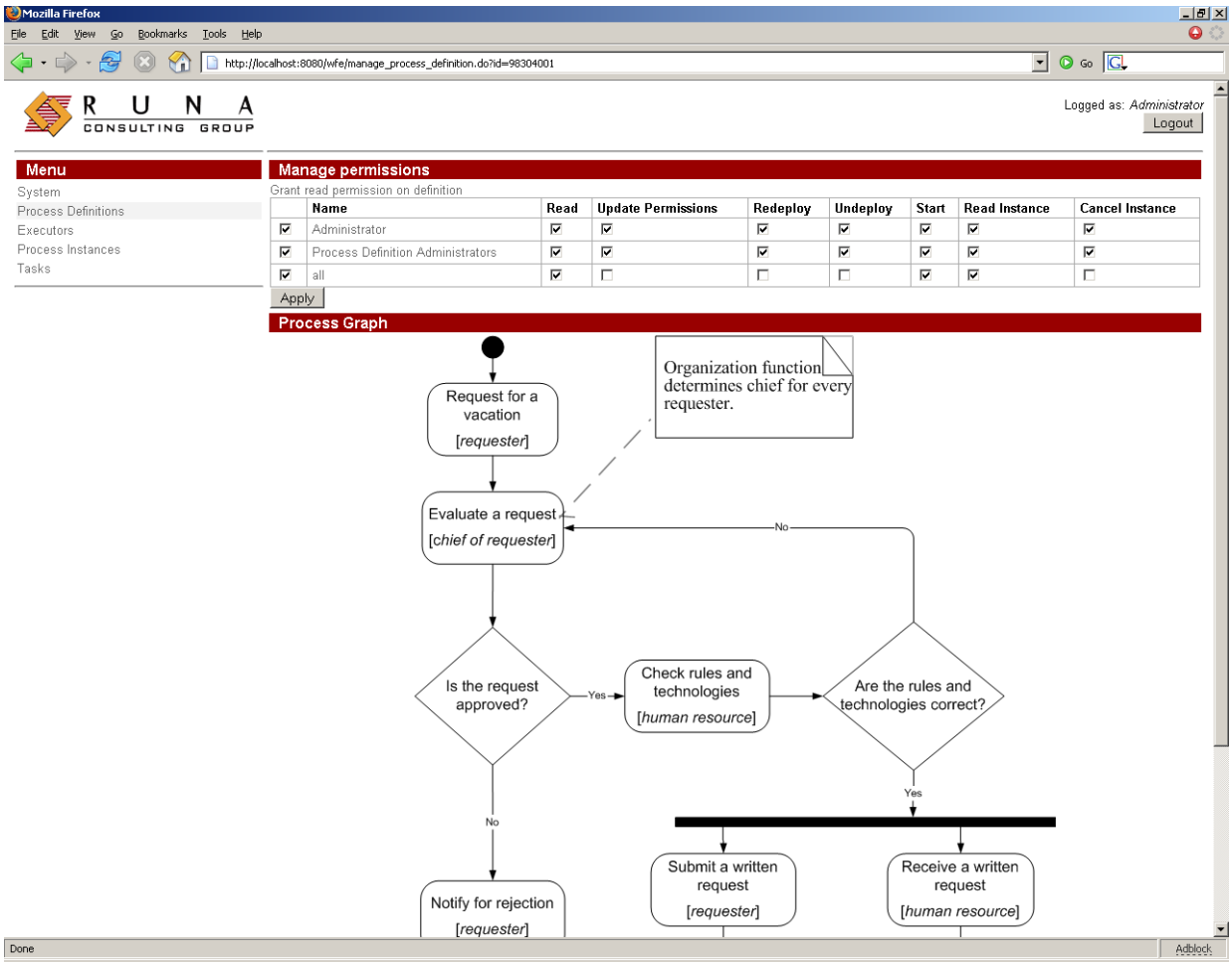


Click on processes “Overtime work”. You’ll see the process properties.
Set the permission “read instance” for *all* group and permissions “read” and “start” for *manager* group.



Do the same for “Business trip” process

Click on processes “Vacation”. Set permissions “read”, “start” and “read instance” for *all* group.



Congratulations, the configuration is ready!!!

AdminKit

AdminKit is a set or administration utilities distributed with RUNA WFE.

Bot Invoker

For a more detailed description of bots, see "Bots Configuration" section of this document.

Description

Bot Invoker is a tool that runs periodic invocation of bots on RUNA WFE Bot Station. Generally Bot Station runs on the same server where RUNA WFE Server installed. For some reasons such as CPU expensive bot tasks or security Bot Station can run on dedicated server.

Command

run
`bot-ivoker.bat <command>` (Windows)
or
`bot-ivoker.sh <command>` (Unix)

Command available:

- start – starts periodic bot invocation
- status – reports bot invocation status (started/stopped)
- stop - stops periodic bot invocation

Returning codes:

- -1 – error
- 0 – bot invoker is not running
- 1 – bot invoker is running

Configuration

To provide bot station location change value of
`ru.runa.bot.delegate.remote.provider.url`
in
`conf\bot_delegate.properties`

To setup invocation period change value of
`invocation.period=30`
in
`server\default\conf\bot_invoker.properties`

Script Runner

Description

Script Runner is responsible for running RUNA WFE scripts.
Scripts are useful for administration. E.g. you can deploy a set of processes with appropriate groups and actors with one click. See scripts/deploy-samples-script.xml.

Command

run
`script-runner.bat` (Windows)
or
`script-runner.sh` (Unix)

Configuration

To provide RUNA WFE Server location change value of
`ru.runa.wfescript.delegate.remote.provider.url`
in
`conf\wfescript_delegate.properties`

Edit script-runner environment to specify:
SCRIPT_PATH="scripts/deploy-samples-script.xml"
LOGIN="Administrator"
PASSWORD="wf"

LDAP Importer

Description

LDAP Importer is used for importing actors and groups from LDAP/MS ACTIVE Directory.

During the import process LDAP importer creates[6] group with name “ldap users”. All imported users and groups will be placed into that group. Members of ldap users group can login into the system and read other users in same group.

Command

run
`ldap-importer.bat` (Windows)
or
`ldap-importer.sh` (Unix)

Configuration

To provide RUNA WFE Server location change value of
`provider.url`
in
`conf\ldap-importer-delegate.properties.properties`

To setup LDAP Server URL, organization units (OU) to search for users and groups, ldap synchronizer principle name and password edit

`runawfe\server\default\conf\ldap-importer.properties`

```
# your LDAP server url
server.url = ldap://172.16.100.2

# your domain
dc = dc=yourdomain,dc=com

# organization units you want to import must be separated by <;>
# if organization unit is missing the synchronization is stopped.
ou = ou=Admins;ou=GOD,ou=Admins;ou=User_policy

# authrorized subject to read directory
principal = cn=LdapReader,ou=User_policy,dc=yourdomain,dc=com
password = YourPr1nc1paLPassw0rd
```

6 If the group wasn't already created.

Bots Configuration

Introduction to Bot

Runa WFE Bot is a program that participates in business processes. Every bot has a link to a Runa WFE actor. Bot executes tasks under the name of this actor. Runa WFE does not distinguish bots from humans.

All bots run inside special environment that is known as WFE Bot Invoker. This application periodically activates all registered bots. Every bot receives tasks assigned to the actor it represents. Then bot passes the tasks to corresponding task handler. When task is performed bot executes activity in workflow process and passes parameters to the process.
Examples of bot task handlers are: generate report, store data to database, send sms, send email, start process, cancel process, write file to disk.
Runa WFE 2.1 sample configuration contains several implemented task handlers:

- E-mailTaskHandler
- DatabaseTaskHandler
- CancelProcessTaskHandler
- SwimlaneAssignerTaskHandler
- UpdatePermissionsTaskHandler

It is always possible to write your own bots using Runa WFE API.

Bot configuration

In order to run bot it is necessary to register it in $(DIST_ROOT)/server/default/conf/bots.xml file.

Bot configuration file structure

Tag bots

Description: bots description root tag. It contains list of registered bots.

Inner tags:

Element	***Short description***
bot	Defines bot

example:

```
<bots xmlns="http://runa.ru/xml" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://runa.ru/xml bots.xsd">
      <bot login="SMTPbot" ...>
...
     </bot>
      <bot login="ExcelBot" ...>
...
     </bot>
      <bot login="HelloBot" ...>
...
```

```
    </bot>
</bots>
```

Tag bot

Description: Defines bot.

Attributes:

Attribute	*Short description*
login	Login of bot
password	Password of bot

Inner tags:

Element	*Short description*
task	Task definition, which bot can perform.

example:

```
    <bot login="ExcelBot" password="123">
        <task name = "overtimes report" .../>
        <task name = "timings report" .../>
        <task name = "worktime report" .../>
    </bot>
```

Tag task

Description: Sets up a correspondence between task and process state name. Assigns task handler and configuration for a task.

Attributes:

Attribute	*Short description*
name	Task name, which bot can perform. The name must be equal to business process state name, which generates the task
handler	Java class name. The class is activated by bot for task performing.
configuration	Configuration file name.

example:

```
<task name = "change data shift" handler =
"ru.runa.wfe.bp.timing.bl.DatabaseTaskHandler" configuration =
"/bot/handler/shift_insert.xml" />


<task name = "overtimes report" handler =
"ru.runa.wfe.bp.commons.ExcelTaskHandler" configuration =
"bot.handler.overtimes_report" />
```

Complete bot configuration example

Consider the following example of bots registration file:

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

<bot login="nero" password="123">

<task name = "Evaluate a request"
handler="ru.runa.wf.logic.bot.DoNothingTaskHandler"
configuration="conf/bot/you_name_it" />

</bot>

</bots>
```

In this configuration there is a single bot, corresponding to actor “nero” with password “123” and this bot can perform task “Evaluate a request” with ru.runa.wf.logic.bot.DoNothingTaskHandler: Bot receives all tasks for executor nero. If it finds a task with name “Evaluate a request”, it reads configuration from conf/bot/you_name_it file and runs ru.runa.wf.logic.bot.DoNothingTaskHandler.

Task handlers

Bot can execute tasks of different types. Every task is associated with task handle. Task handler is Java class that that must implements interface ru.runa.wf.logic.TaskHandler. The interface has two methods:

- handle(Subject subject, TaskStub taskStub) – task performing
- configure(String configurationName) – configuration reading

Runa WFE has build in set of task handlers. They are:

- DoNothingTaskHandler – prints text on screen of server (desingned for educational and simulation purpose)
- DatabaseTaskHandler – performs operations with Databases
- StoreDataToDatabaseTaskHandler – stores defined process variables to database
- LoadDataFromDatabaseTaskHandler – loads defined process variables from database
- EmailTaskHandler – sends email to email address in HTML format
- TextEmailTaskHandler – sends email to email address in text format

DatabaseTaskHandler

Database task handler uses *.xml configuration files. The basic element of the scheme is tag “task”. The tag attribute datasource defines JDBC datasource connection.
Note: URL must contain user name and password for database connection.

Database task handler configuration example:

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

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

<task datasource="java:/DefaultDS">

<queries>

<query sql="update TIMINGS set CONFIRMATION_DATE = ? where INSTANCE_ID = ?">
                                            <param var="currentDate" />
                    <param var="instanceId" />

</query>
```

```
</queries>
</task>
</database-tasks>
```

E-mailTaskHandler

E-mail task handler uses *.properties configuration files. Files have Latin 1 encoding.
List of E-mail task handler properties:

- smtp.server – SMTP server name or IP address.
- smtp.user – user name
- smtp.password – user password
- subject – e-mail subject
- content.type – message content type, default is text/plain. (see. http://www.ietf.org/rfc/rfc2045.txt http://www.ietf.org/rfc/rfc2046.txt)
- to – recipient address
- from – sender address
- reply.to – «Reply-To» heading
- cc – «cc» heading
- bcc – «BCC» heading

The e-mail body will contain the corresponding task form.

E-mail task handler configuration example:

```
smtp.server=my_smtp_server.my_domen.com
from=bot_account@my_domen.com
subject=<customtag var="employee" delegation =
"ru.runa.wf.web.html.vartag.ActorFullNameDisplayVarTag" />
to=recipient@my_domen.com
```

CancelThisProcessInstanceTaskHandler

There are situations you need to cancel the process without waiting for other tasks to be done.

In this situation CancelThisProcessInstanceTaskHandler is your friend.

It doesn't need any configuration. Once it receives a task it cancels process instance task belongs to.

MSWordReportTaskHandler

Uses MS Word document as template. Bookmarks are placeholders for variables.

Variables can be formatted with Format class specified in task handler configuration.

After handling document is stored as process file variable.

Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
<msword-report-task xmlns="http://runa.ru/xml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://runa.ru/xml msword-report-task.xsd">
      <!--  Bookmarks in sample-template.doc will be replaced by formatted
values of process variables.
            Sample-templated will be saved as report.doc  and assigned to
process variable resultWord.  -->
      <report template-path="c:\template\sample-template.doc" output-
```

```
variable="resultWord"  output-variable-file-name="report.doc">
            <mapping bookmark="dateInYearMonthDayFormat" variable="date"
format-class="java.text.SimpleDateFormat" format="yyyy.MM.dd" />
            <mapping bookmark="dateInDayMonthFormat" variable="date" format-
class="java.text.SimpleDateFormat" format="DD.MM" />
            <mapping bookmark="doubleWithDefaultFormat" variable="double"
format-class="java.text.DecimalFormat" />
            <mapping bookmark="doubleFormated" variable="double" format-
class="java.text.DecimalFormat" format="#,##" />
            <!-- If format-class is omitted toString method is called on
variable value. -->
            <mapping bookmark="string" variable="string" />
      </report>
</msword-report-task>
```

For sample process see samples\MS Word Report

Running bots

See "AdminKit/Bot Invoker" section of this document for an explanation on how to run bots.

Appendix A. GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, “this License” refers to version 3 of the GNU Lesser General Public License, and the “GNU GPL” refers to version 3 of the GNU General Public License.

“The Library” refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An “Application” is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A “Combined Work” is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the “Linked Version”.

The “Minimal Corresponding Source” for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The “Corresponding Application Code” for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
    - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
    - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
- e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany

the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.