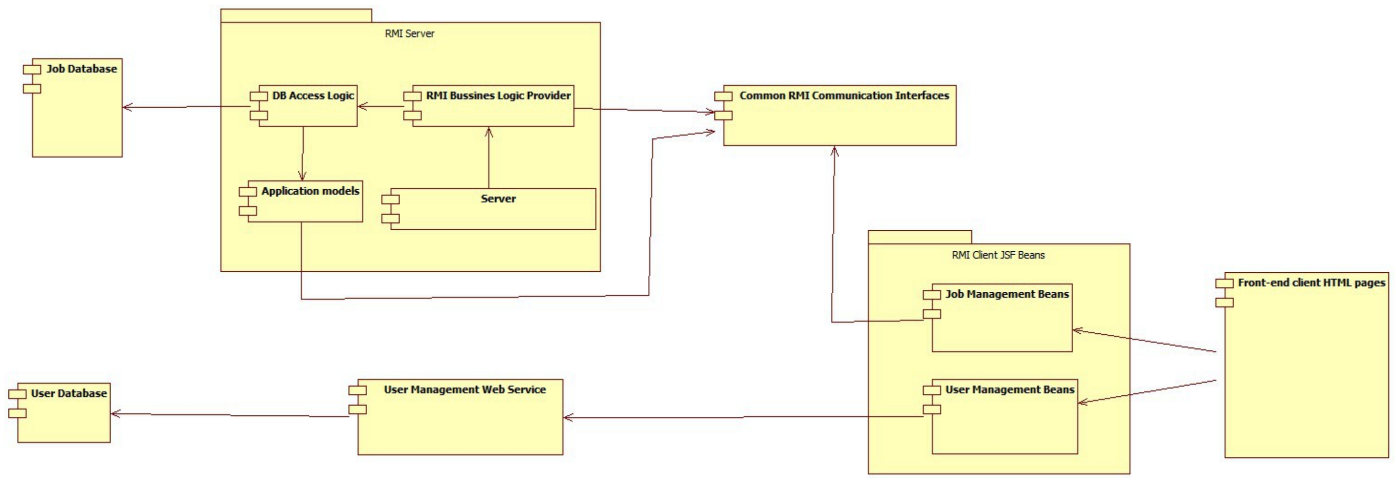
Distributed Systems – assign 3

-Stefan Prisca, group 30441-

Conceptual Diagram

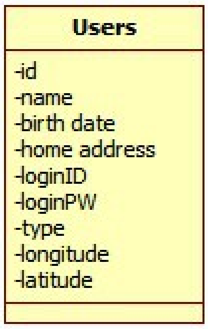


In the above diagram, you can clearly see all the important parts of this application and how they interect with each other:

1. The Job Database: is the database used to store information about jobs, which also includes job categories
2. The RMI Server component: This includes all components that run as part of the RMI server:
    1. The DB Access Logic: this is the persistence API and extra bussines logic that is used to access the DB.
       For this application, Eclipse Link was chosen as the persistence API. It offers a wide range of possible database connections (including SQL Dbs and NonSql dbs). It is also easy to configure through xml files, and has a central maven repository that facilitates the maven build.
    2. Application Models: These are in-memory representations of the data models from the database.
    3. The RMI Bussines Logic Provider is responsible for providing bussines logic to the client. This contains the class that implements the common communication interface and that will be exported to the client at runtime.
    4. Server: this is the actual RMI server that will register the logic provider under a given address.
3. Common RMI Communication Interfaces: these are the interfaces used to communicate between the RMI server and the RNI client.
4. The RMI Client JSF Beans package includes the JSF beans that will act as the RMI client in this applicaiton, as well as the beans that will deal with user management.
5. The User Management Web Service is a SOAP web service that handles user management requests like login validation, registering a new user, etc.
   This is separate from the rest of the RMI application, and is a stand-alone web service that uses it's own user database.
6. Client-side HTML pages: contains the web pages that will be displayed in the browser. These were build using the JSF2 technologies.

Data Design

Users

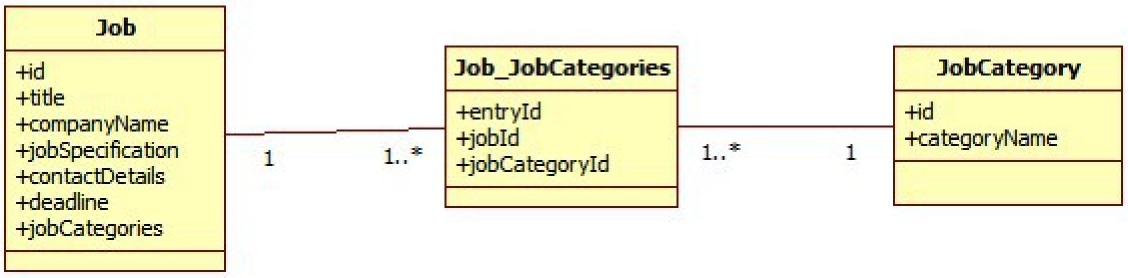


First data element consists of the table that contains all the user information that is required for the application to run:

- id: database identifier for the current user
- name: name of the user
- birth date: the birth date of the user
- home address: the home address of the user.
- LoginID: the id the user is using to log in the application
- LoginPW: the password used to log in to the application
- type: the type of the user: administrator or regular user.
- logitude and latitude: world coordinates for the user to determine timezone.

This user information is provided via the web service.

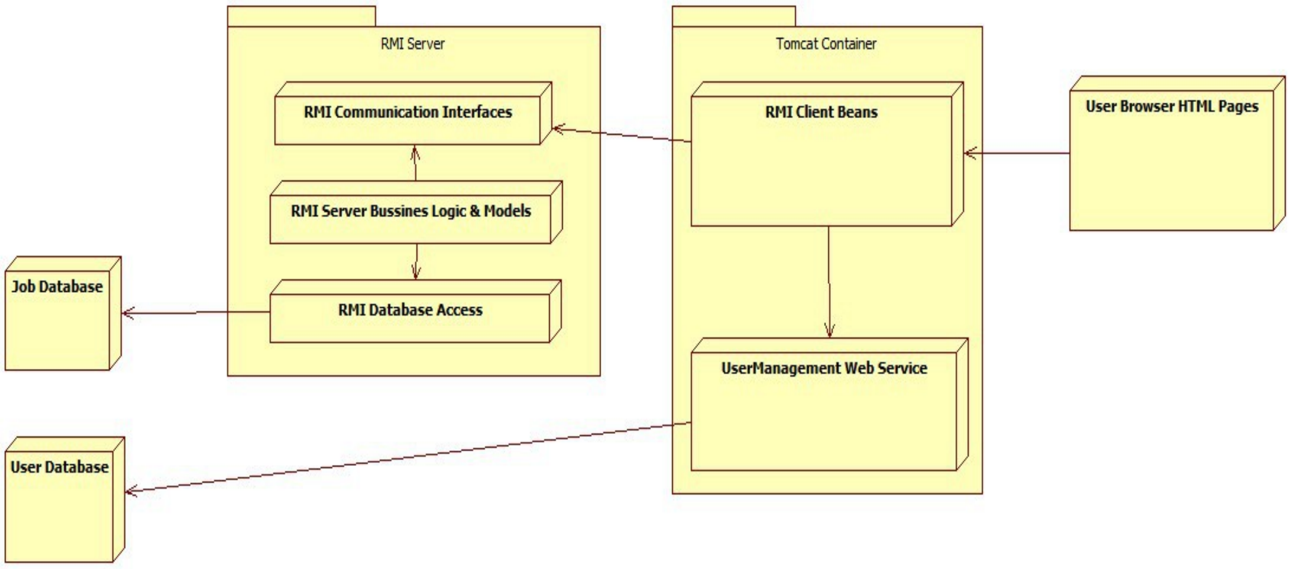
Jobs



The job information is split between three tables:

1. Job table: contains the basic information about jobs, like:
   title, company name, specification, contact details, deadline and job categories
2. Job_JobCategories table contains a mapping from jobs to job categories. As each job can have multiple categories and each category can belong to multiple jobs, this table is required
3. JobCategory table contains the information about job categories, like category name.

Deployment Diagram



The above diagram shows how the components will be deployment into the running application:

1. The Job Database will be separate from the application, and can be hosted on any Micrisoft SQL server.
2. The RMI server will contain all bussines login required, including the data base access and models. This will also host the common interfaces that are used to communicate with the RMI client.
3. The RMI Client Beans (Including the UserManagement beans) will be deployed in the Tomcat container. As these are developed as a web application, the logic will be available on the entire network.
4. The Tomcat container will also include the User Management Web Service. This service will be available as an SOAP service.
5. The User Database is separate from the application, and can be hosted on any Microsoft SQL Server.
6. The User HTML pages will be displayed in the user's browser.

Installing the application

It is recommended to use maven tool for building and deploying the project.
All the pom files are configured, so in order to run the build, do the following:

1. Clone the project git repository
2. Go to the App3 folder.
   There should be three projects here:
   1. ro.stefanprisca.distsystems.app3.client
      This is the JSF RMI client project
   2. ro.stefanprisca.distsystems.app3.common
      This project will contain the common interfaces for RMI communication
   3. ro.stefanprisca.distsystems.app3.server
      This project represents the RMI server, and contains bussines logic as well as database access logic.
3. Before installing the application, you'll have to install the mssql jdbc driver and the web service for user management:
   1. MSSQL JDBC driver:
      1. go to *git_root/utils/mssqljdbc* folder
      2. you should have the *sqljdbc4.jar* driver there.
      3. For windows systems, there is a *mvn_install.bat* file that will install this driver using maven.
      4. For other systems, use the command:
         mvn install:install-file -Dfile=sqljdbc4.jar -DgroupId=com.microsoft.sqlserver -DartifactId=sqljdbc4 -Dversion=4.0 -Dpackaging=jar
   2. UserManagement Web Service:
      You need this installed in the local maven repository because the RMI client project has a dependency on it, as it is required to use the objects provided by the service.
      1. go to *git_root/utils/usermanagement* folder
      2. you should have the *ro.stefanprisca.distsystems.utils.login.jar* file there.
      3. For windows systems, there is a *mvn_install.bat* file that will install this driver using maven.
      4. For other systems, use the command:
         mvn install:install-file -Dfile=ro.stefanprisca.distsystems.utils.login.jar -DgroupId=ro.stefanprisca.distsystems -DartifactId=ro.stefanprisca.distsystems.utils.login -Dversion=0.1.0 -Dpackaging=jar -DgeneratePom=true
4. After maven has installed the jdbc driver and the usermanagement utils in your local repository, go back to the application folder (App3) and run *mvn clean install*
   Note: before you do this, you should run the *App3Server.java* main method first. This will start the RMI server, and there are some tests in the build that will require this to be running. You can do this by opening the *ro.stefanprisca.distsystems.app3.server* project in Eclipse and then simply run the *App3Server.java* class.
   If not, you can also add *-Dmaven.test.skip=true* at the end of the install command in order to skip the test:
   mvn clean install -Dmaven.test.skip=true
5. This will install the application, and generate the *ro.stefanprisca.distsystems.app3.client* file in ..\App3\ro.stefanprisca.distsystems.app3.client\target

Deploying the application using Tomcat.

In order to deploy the application, you'll have to install Tomcat as a local service first.

After you did this, copy the above mentioned *ro.stefanprisca.distsystems.app3.client.war* and the *utils/usermanager/ro.stefanprisca.distsystems.utils.login.war* files in the webapp folder of Tomcat.

After you do this, make sure the the *App3Server* is running (either from Eclipse or from a separate jvm started from the command line). Without this, you will get some complaints about bean initializations, as the application will not be able to access the remote objects.

Now you just have to start Tomcat and access the page:
http://localhost:xxxx/ro.stefanprisca.distsystems.app3.client/home.xhtml

Further instructions on how to deploy applications with apache tomcat can be found here.

Opening the application in Eclipse:

The App3 projects:
If you have the Eclipse Maven plug-in installed, all you need to do is import the projects as Maven projects.
Note that this is the recommended way to import the projects in Eclipse.

If not, you'll have to select *Create New Project,* deselect the *Use default location* option, and set the project location to the *ro.stefanprisca.distsystems.app3.** folder.

The Usermanagement web service:

This project does not (yet) have a maven infrastructure. You will have to import this from eclipse, using the method presented above (by creating a new project in that file location).
Note that after you create the project, you'll have to make sure to import all the required libraries.

Further note: you don't really need to open this project. If you want to run the application, just copy the war file in Tomcat, and the service will perform as expected (tests have been made :-).

Running the application.

Before starting the application, you'll need to also configure two local databases, for the users and the jobs. You will find the specifications for this data bases in the *persistence.xml* files:

- Users: *\utils\ro.stefanprisca.distsystems.utils.login\src\META-INF\persistence.xml*
- Jobs: *\App3\ro.stefanprisca.distsystems.app3.server\src\main\resources\META-INF\persistence.xml*

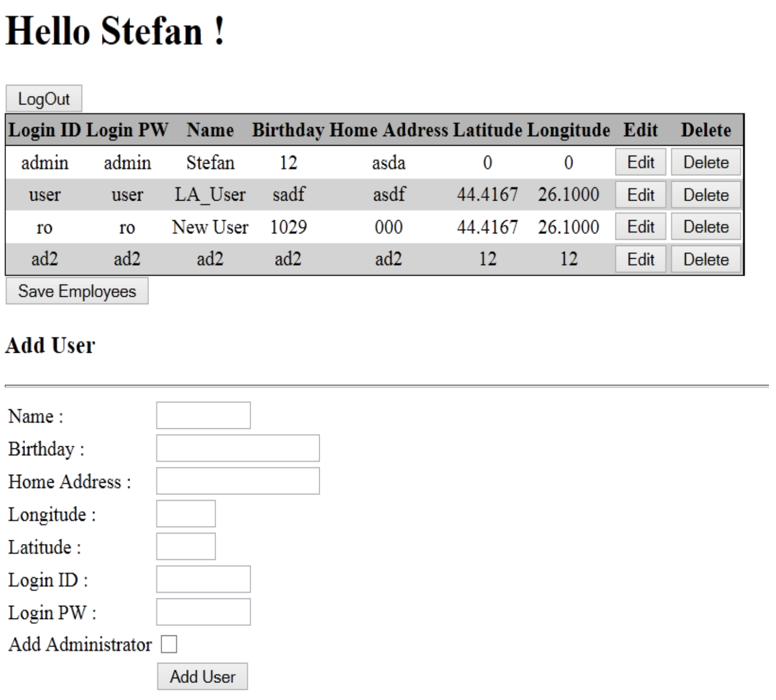
Also, you might want to add some accounts in the Users database in order to log in. I usually use the following two:

1. administrator account: id = admin; pw = admin
   Use this account to test the administrator functions
2. regular user account: id = user; pw = user
   Use this acount to test the regular user functions.

After you deploy the application using tomcat, and you access the application through the browser, you'll have to log in.

Depending on your log in credentials, you will be redirected to the administrator page or to the regular user page.
Note: If you are not logged in as an administrator, and you try to access the administrator page, you will be redirected to the log in page. Only logged in administrators can access this page!

1. Administrator page:
   Here you will see a table with all the user records from the database.
   You can edit users information, delete users or add aditional users to the data base.
   Note that after you edit user entries, you'll have to press the *Save Users* button in order to persist the changes in the data base.
   If you edit the longitude and latitude fields, it is highly recommended to use real world coordinates for them. This will allow you to further test the Timezone recognition functionality.



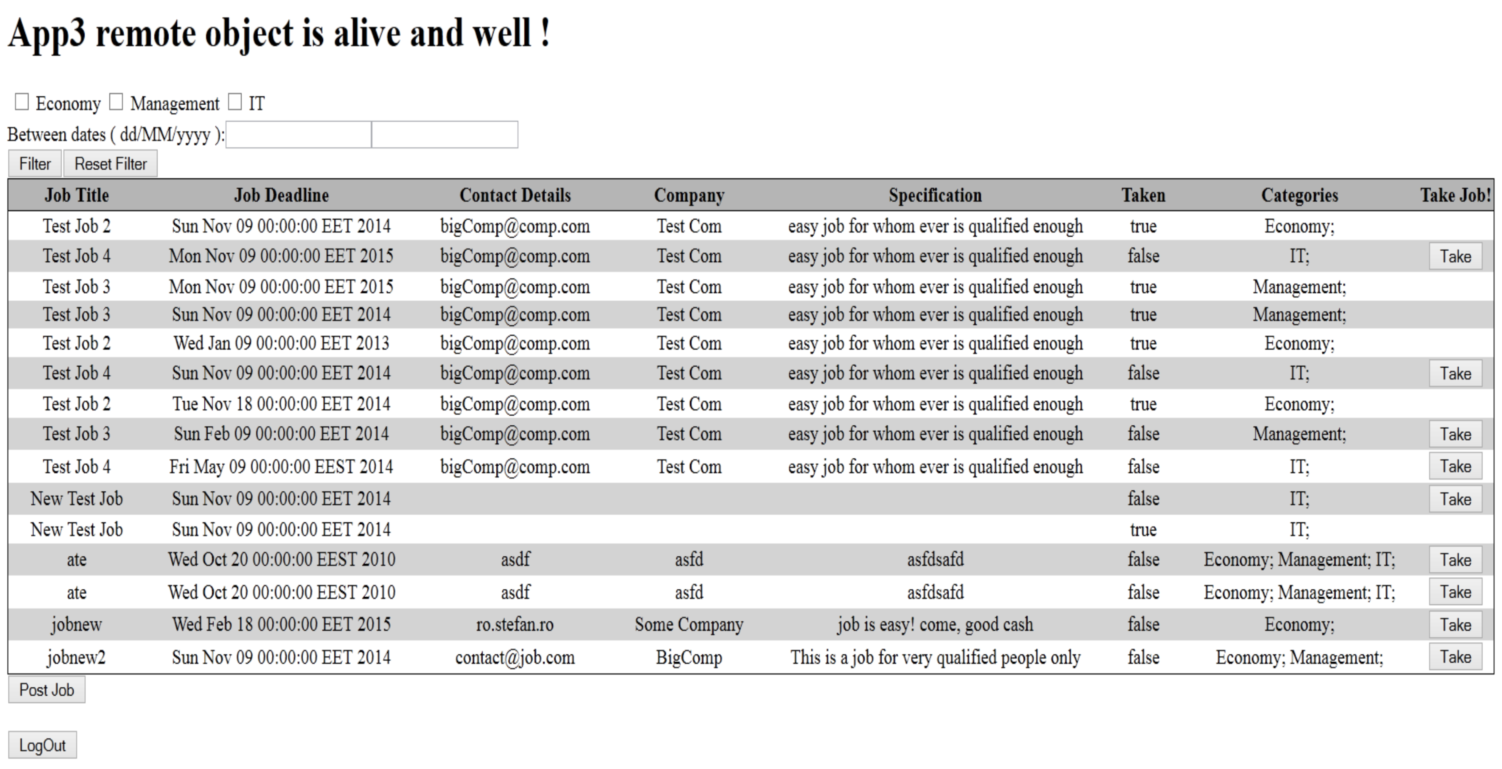
Hello Stefan !

Login ID	Login PW	Name	Birthday	Home Address	Latitude	Longitude	Edit	Delete
admin	admin	Stefan	12	asda	0	0	Edit	Delete
user	user	LA_User	sadf	asdf	44.4167	26.1000	Edit	Delete
ro	ro	New User	1029	000	44.4167	26.1000	Edit	Delete
ad2	ad2	ad2	ad2	ad2	12	12	Edit	Delete

Add User

Name :	
Birthday :	
Home Address :	
Longitude :	
Latitude :	
Login ID :	
Login PW :	
Add Administrator	☐

2. Regular user page:
   Here you will find job offers and aditional informations.
   Furthermore, you can:
   1. filter the job offers by using the Category and Start/End Date filters
   2. Take a job that you are interested in by pressing the Take button on the far right of each job entry.
      Note that this button will not be displayed if the job is allready taken.
   3. Post new jobs by using the Post Job button at the end of the table.

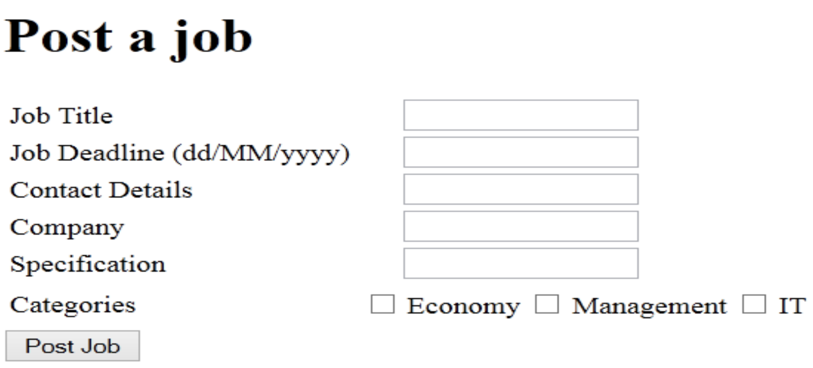


App3 remote object is alive and well !

☐ Economy ☐ Management ☐ IT

Between dates (dd/MM/yyyy):

Job Title	Job Deadline	Contact Details	Company	Specification	Taken	Categories	Take Job!
Test Job 2	Sun Nov 09 00:00:00 EET 2014	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	true	Economy;	
Test Job 4	Mon Nov 09 00:00:00 EET 2015	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	false	IT;	Take
Test Job 3	Mon Nov 09 00:00:00 EET 2015	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	true	Management;	
Test Job 3	Sun Nov 09 00:00:00 EET 2014	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	true	Management;	
Test Job 2	Wed Jan 09 00:00:00 EET 2013	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	true	Economy;	
Test Job 4	Sun Nov 09 00:00:00 EET 2014	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	false	IT;	Take
Test Job 2	Tue Nov 18 00:00:00 EET 2014	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	true	Economy;	
Test Job 3	Sun Feb 09 00:00:00 EET 2014	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	false	Management;	Take
Test Job 4	Fri May 09 00:00:00 EEST 2014	bigComp@comp.com	Test Com	easy job for whom ever is qualified enough	false	IT;	Take
New Test Job	Sun Nov 09 00:00:00 EET 2014				false	IT;	Take
New Test Job	Sun Nov 09 00:00:00 EET 2014				true	IT;	
ate	Wed Oct 20 00:00:00 EEST 2010	asdf	asfd	asfdsafd	false	Economy; Management; IT;	Take
ate	Wed Oct 20 00:00:00 EEST 2010	asdf	asfd	asfdsafd	false	Economy; Management; IT;	Take
jobnew	Wed Feb 18 00:00:00 EET 2015	ro.stefan.ro	Some Company	job is easy! come, good cash	false	Economy;	Take
jobnew2	Sun Nov 09 00:00:00 EET 2014	contact@job.com	BigComp	This is a job for very qualified people only	false	Economy; Management;	Take



Post a job

Job Title	
Job Deadline (dd/MM/yyyy)	
Contact Details	
Company	
Specification	
Categories	☐ Economy ☐ Management ☐ IT

References

- JSF2: <http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html>
- EclipseLink: <https://www.eclipse.org/eclipselink/>
- Tomcat: <http://tomcat.apache.org/>
- Maven: <http://maven.apache.org/>
- Project github repo: <https://github.com/stefanprisca/FacultDistributedSystems>