Universidad de Oviedo

School of Computer Science

GestUsers: User Management System

CITIZEN PARTICIPATION

Software Architecture for GestUsers.
Description of the practice work (2017)

Description of the practice work done by the work teams of the course "Software Architecture" during the academic year 2016-17.

School of Computer Science Engineering

2017-06-26

GRADO DE INGENIERÍA INFORMÁTICA DEL SOFTWARE

GestUsers: User Management System

Authors:

Aquilino Adolfo Juan Fuente

PhD. Computer Science
Dept. Computer Science, University of Oviedo

Jose Emilio Labra Gayo

PhD. Computer Science
Dept. Computer Science, University of Oviedo

Juan Luis Mateo Cerdán

PhD. Computer Science
Dept. Computer Science, University of Oviedo

Katia Fernández Fernández

Student of Software Engineering at Universidad de Oviedo

Andrei Manu

Student of Software Engineering at Universidad de Oviedo

Christian Martínez Abad

Student of Software Engineering at Universidad de Oviedo

Date: 26/06/2017

Version: 0.06

Table of Contents

1 Introduction and Goals

The goal of this document is to describe the structure of an architecture of User Management that will be reused. Although the system that we describe has its own functionality, the main goal is that it will be part of a general system of citizen participation.

This document describes the laboratory work of the course "*Software Architecture*" which is taught by the three first authors. The course is part of the Degree in Software Engineering, School of Computer Science Engineering, University of Oviedo.

The system is divided in four different parts: CitizensLoader, to load data about citizens; Participants, to check if a user can participate; ParticipationSystem, to configure different parameters in order to allow the required actions; Dashboard, to view the evolution of the participation system in real time.

The development of the hole system will be done by a group of three people.

2 Requirements

User Management will be divided in four parts:

- CitizenLoader: loads the list of users from the Council, for example the municipal census.
- Participants: checks if a citizen can participate.
- ParticipationSystem: manages the citizen participation and allows to configure different parameters in order to do the required actions.
- Dashboard: offers a dashboard to the Council staff, councilmen and other authorities in order to view the evolution of the participation system in real time.

2.1 CitizensLoader

The System administrator must be able to introduce data from the citizens list. That data can be obtained from different sources like the municipal census, lists of immigrants without official residence, bystanders, etc. Those lists will be delivered by some institution to the Council.

The introduction of data will be made from Excel files that contain a list of rows with the following information:

- First name
- Last name
- Email
- Date of birth
- Address
- Nationality
- ID (National ID, the residence card ID, etc.)

When importing the citizens data, the system will create a user (whose login name will be the email) and a random password which will enable the user to enter the system to check if the data is correct as well as to later participate in the system. The system will generate personal letters that will be sent to each user by email. This task will be done by the Council and is not part of this system.

If a user appears in two different lists, this event will be recorded and informed in a log file. A user can only be created once. If the data is different from the current data available in the system, the current data will not be modified and an error will be recorded in the log.

The system is extended in order to emit the letters in two different formats: PDF and txt.

All the errors detected in the input file, will be reported in a log file.

2.2 Participants

Citizens should be able to login into the system to check that they can participate once the notification letter has been received. In order to implement that feature, a simple web service will be created that has two parameters passed as a POST message: login name and password and returns the data available about the citizen if the information is correct or reports an error if it isn't. Both the call parameters and the return information will employ JSON format.

The web service offers a simple HTML interface where a user can login and see his information in a human- friendly way.

2.3 ParticipationSystem

The personnel from the Council must be able to configure the participation system in order to configure different parameters like proposal categories, dates in which they will be active or not allowed words as other kind of issues.

The popularity of each proposal comes from the votes of other users. A proposal will be accepted only in case it is supported by other users. A proposal is considered accepted if is satisfies the minimal number of users supporting it. This number is configured by the portal administrators.

Once a proposal enters the acceptance phase (it has at least the minimum number of supporters), the portal administrators receive a notification of this event.

A proposal that has passed the acceptance phase could be updated by the votes that result from the corresponding parliament.

Users can also comment proposals in order to generate some discussion about them and also improve them. These comments can be ordered by chronological order.

The different proposals, supports and comments that they obtain will be recorded in the application log which will be connected to a Kafka Stream.

The system is extended in order to allow users vote the comments. [Optional done]

2.4 Dashboard

The Council staff, councilmen and other authorities will have the possibility to view the evolution of the participation system in real time. They can see graphically the key performance indicators that affects to the participation system which are constantly updated.

Those key performance indicators are:

- Show the changes that appear in the participation system as they are produced.
- Update information in all of the concurrent clients that are connected at update time.
- Offer information about the evolution of the proposals (the number of votes that they receive as well as the different comments).

3 Methodology

This document employs the ADDttribute-Driven Design) methodology (Bass, Clements, & Kazman, 2003)and the SEI (ANSI/IEEE 1471, 2000).

The templates have also been inspired by the Arc42 templates (http://arc42.org/) where documentation architecture templates are defined in English, German and Spanish.

Another project that follows those templates for a biking domain is available at:

http://biking.michael-simons.eu/docs/index.html

4 Stakeholders

The stakeholders identified are:

1. Students that develop the assignment
2. System Administrator
3. Citizens
4. Developers of the system
5. Course Teachers
6. Council staff, councilmen and other authorities

Code	Stakeholder	Interests (Modules)
ST-01	Students	All of them
ST-02	System administrator	Load files
ST-03	Citizens	Check data and participate
ST-04	Developers of Participation System	Check data, implement the configuration options of the system and a Kafka Stream
ST-05	Course Teachers	All of them
ST-06	Council staff, councilmen and other authorities	Configuration and view evolution of the participation system

Table 1. List of stakeholders/interests

4.1 Students that develop the assignment

This group is formed by the team that will develop the system. Some of their goals are:

- Use of known technologies and methodologies minimizing the risks to learn new ones.
- Learn how to develop software collaboratively and in a professional way.
- Use of simple technologies that can interoperate with other systems.

4.2 System administrator

This is the person who is in charge of loading the citizens list.

Some of the goals are:

- Use of simple and well-known technologies for input files.
- Files that can be read by humans.
- Be able to automate the loading process.
- Be able to debug the loading process in case of failures.

4.3 Citizens

These are the final users of the system. Some of their goals are:

- Get access to the system in a simple way.
- Being able to participate from their homes in a safe way.
- Being able to query their status in the system.
- Being able publish a proposal and comment and vote all of them.

4.4 Developers of the System

In this case, the team that will implement the system is the same for all of the modules, so because of this there is no clear distinction between the developers of the different modules of the system.

Some of their goals are:

- Have a simple way to detect if a citizen can participate in the system as soon as possible.
- Use of simple technologies that can interoperate with other systems.
- Implement the configuration options of the system.
- Offer to the Council staff, councilmen and other authorities the possibility to see the statistics and the evolution of the participation system.
- Implement a Kafka Stream.

4.5 Course teachers

They are responsible for the results of this assignment. Some of their goals are:

- Use technologies that help students acquire skills related with Software Architecture by developing a practical assignment.
- Introduce the students in collaborative and professional software development through TDD (Test driven development) techniques.
- Show the students an example documentation of a software architecture.

4.6 Council staff, councilmen and other authorities

This is the team responsible of the configuration of the participation system. Some of their goals are:

- Configure the different options of the participation system, such as proposal categories, dates in which they will be active, not allowed words, minimal number of support votes...
- View in a friendly way the evolution of the participation system in real time. Each type of people will be able to see a different type of visualization of the system.

5 Quality Attributes

We have identified the following quality attributes:

- **Availability**
  - The system must be able to process data 24x7.
  - The system must be able to serve thousands of requests at the same time.
  - The system will show in real time the results.
- **Modifiability**
  - Easily change some parts of the application: Add an error reporting feature.
  - Easily modify some parts of the application: Add other output files to generate the letters.
  - Easily modify the configuration of the system: Enable different options to be configured.
  - Easily modify some parts of the application: Restrict who and how the dashboard will be seen.
- **Performance**
  - The performance of the data loading system is reasonable.
  - Querying information about a user through the web service should be fast.
  - The system must be able to process a big workload without breaking.
  - The dashboard reflects the changes in participation system as they are being produced.
- **Security**
  - The system should warrant the confidentiality of the citizens' data.
  - The system must warrant the confidentiality of the votes.
  - The system must warrant that only allowed people have access to configure the system.
  - The system should warrant that the dashboard can only be viewed by the authorized people.
- **Testability**
  - It must be testable that the citizens' data loading process is correct.
  - It must be testable that the web service behaves as expected.
  - It must be testable that the Kafka Stream shows the statistics properly. It will be tested independently with a small simulator that generates random events.
- **Usability**
  - The data loading system must be easy to use by System administrator users which are familiar with Unix-like tools.
  - The system must be easy to use by all the users and allow them to do what they want in a simple way.
  - The results shown have to be clear and impossible to misunderstand.
- **Interoperability**
  - This system will be used by the Participation System which will leverage on it for user management. The Participants web service must be used by an automated process that can query the status of a user.
  - The results must be updated dynamically and in real time without user interaction.
- **Simplicity**

    - The four modules should be simple and easy to develop.
- **Deployability**
    - The system should be easily deployable, especially in a cloud based server.
- **Scalability**
    - The system will be able to show the data of the users even if it has more than a thousand users.

5.1 List of Quality Attributes

The list of quality attribute is the following:

Code	Description	Type of Attribute	Module
AT001	The system must be able to process data 24x7	Availability	Participants, Participation system
AT002	The system must be able to serve millions of requests at the same time	Availability	ParticipationSystem
AT003	The system will show in real time the results	Availability	Dashboard
AT004	Easily change some parts of the application: Add an error reporting feature	Modifiability	CitizensLoader
AT005	Easily modify some parts of the application: Add other output files to generate the letters	Modifiability	CitizensLoader
AT006	Easily modify the configuration of the system: Enable different options to be configured	Modifiability	ParticipationSystem
AT007	Easily modify some parts of the application: Restrict for whom and how the dashboard will be seen	Modifiability	Dashboad
AT008	The performance of the data loading system is reasonable	Performance	CitizensLoader
AT009	Querying information about a user through the web service should be fast	Performance	Participants, ParticipationSystem
AT010	The system must be able to process a big workload without breaking	Performance	ParticipationSystem
AT011	The dashboard reflects the changes in participation system as they are being produced	Performance	Dashboard
AT012	The system should warrant the confidentiality of the citizens' data	Security	All of them
AT013	The system must warrant the confidentiality of the votes	Security	ParticipationSystem, Dashboard
AT014	The system must warrant that only allowed people have access to configure the system	Security	ParticipationSystem
AT015	The system should warrant that the dashboard can only be viewed by the authorized people	Security	Dashboard
AT016	It must be testable that the citizens' data loading process is correct	Testability	CitizensLoader

Code	Description	Type of Attribute	Module
AT017	It must be testable that the web service behaves as expected	Testability	Participants, ParticipationSystem
AT018	It must be testable that the Kafka Stream shows the statistics properly. It will be testes independently with a small simulator that generates random events	Testability	ParticipationSystem, Dashboard
AT019	The data loading system must be easy to use by System administrator users which are familiar with Unix-like tools	Usability	CitizensLoader
AT020	The system must be easy to use by all the users and allow them to do what they want in a simple way	Usability	All of them
AT021	The results shown have to be clear and impossible to misunderstand	Usability	Dashboard
AT022	This system will be used by the Participation System which will leverage on it for user management. The Participants web service must be used by an automated process that can query the status of a user	Interoperability	Participants
AT023	The results must be updated dynamically and in real time without user interaction	Interoperability	Dashboard
AT024	The four modules should be simple and easy to develop	Simplicity	All of them
AT025	The system should be easily deployable, especially in a cloud based server	Deployability	All of them
AT026	The system will be able to show the data of the users even if the number of users reaches 5 million	Scalability	ParticipationSystem, Dashboard

Table 2. List of quality attributes and their types

5.2 Quality Attributes and stakeholders

The following table shows which attribute qualities are interesting for which stakeholder:

Attributes vs Stakeholders	ST-01	ST-02	ST-03	ST-04	ST-05	ST-06
AT001	X		X	X		X
AT002	X		X	X		X
AT003	X		X	X		X
AT004	X			X		
AT005	X		X	X		
AT006	X			X		X
AT007	X		X	X		X
AT008	X		X	X	X	X
AT009	X		X	X		X
AT010	X			X	X	
AT011	X		X	X		

Attributes vs Stakeholders	ST-01	ST-02	ST-03	ST-04	ST-05	ST-06
AT012	X		X	X		
AT013	X	X	X	X		X
AT014	X		X	X		X
AT015	X		X	X	X	X
AT016	X	X		X	X	
AT017	X		X	X		X
AT018	X			X	X	
AT019	X	X		X		
AT020	X	X	X	X	X	X
AT021	X		X	X	X	X
AT022	X		X	X		X
AT023	X			X	X	
AT024	X			X	X	
AT025	X			X	X	
AT026	X			X	X	

Table 3. List of stakeholders: interests vs quality attributes

6 Architecture Constraints

6.1 Technical constraints

We have detected the following set of technical constraints in the project:

Code	Constraint	Background/Motivation
TC001	All the systems will be implemented in Java	The developer team (ST001) has knowledge of Java
TC002	The data will be stored in a relational database.	The developer team (ST001) has knowledge of relational databases and there are a lot of libraries to work with relational databases from Java
TC003	The web service will be based on REST using JSON format	The REST style of web services using JSON is very popular and easy to implement nowadays.
TC004	The input data format to load data is Excel	Excel is a popular format for data exchange and there are several libraries to process Excel files
TC005	The output data of the citizens loader module will be a set of text files	In order to facilitate the implementation, text files are the easier format to generate. However, the developer team can optionally implement other generators.
TC007	Automated testing	The tests should be run automatically and a continuous integration system must be used
TC008	The web service will be implemented using the Spring Boot web framework	Spring Boot web framework leverages on Spring, which is a well-known framework very popular in Industry. It contains lots of examples and help info that can help students to learn to use it.
TC009	Use of Kafka Stream	Kafka is used for building real-time data pipelines and streaming apps. For this reason is a perfect solution to publish log data that will be updated in real time to the subscribers

Table 4. Technical constraints

6.2 Organizational Constraints

Code	Constraint	Background/Motivation
OC001	The system will be implemented by a small team of student developers.	The size of the teams will be of 3 students. The goal is that students learn to work collaboratively by developing a simple project
OC002	The source code will be available as a github repository	Github offers a very powerful project management tool for this kind of projects.

Table 5. Organizational constraints

7 System scope and context

The system is decomposed in four modules:

- CitizensLoader: This module will be responsible to convert data from Excel files and load it into the database. The system will be invoked by a system administrator.
- Participants: This module will check if users can participate obtaining information from the database.
- ParticipationSystem: This module will be in charge of managing citizen participation. It will allow to configure different parameters like proposal categories, dates in which they will be active, not allowed words and other kind of issues.
- Dashboard: This module will offer a dashboard with statistics from the ParticipationSystem.

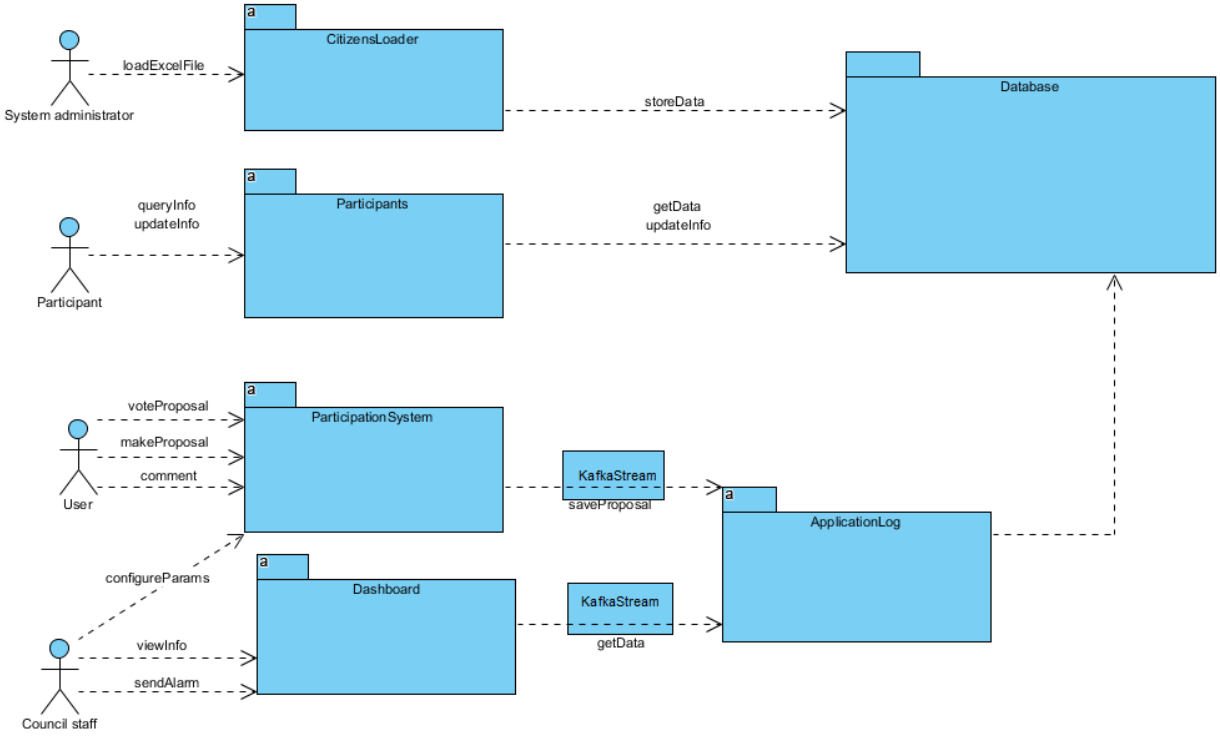

Figure 1. Business Context

The following figure contains a BPMN diagram showing the whole process of all the system.

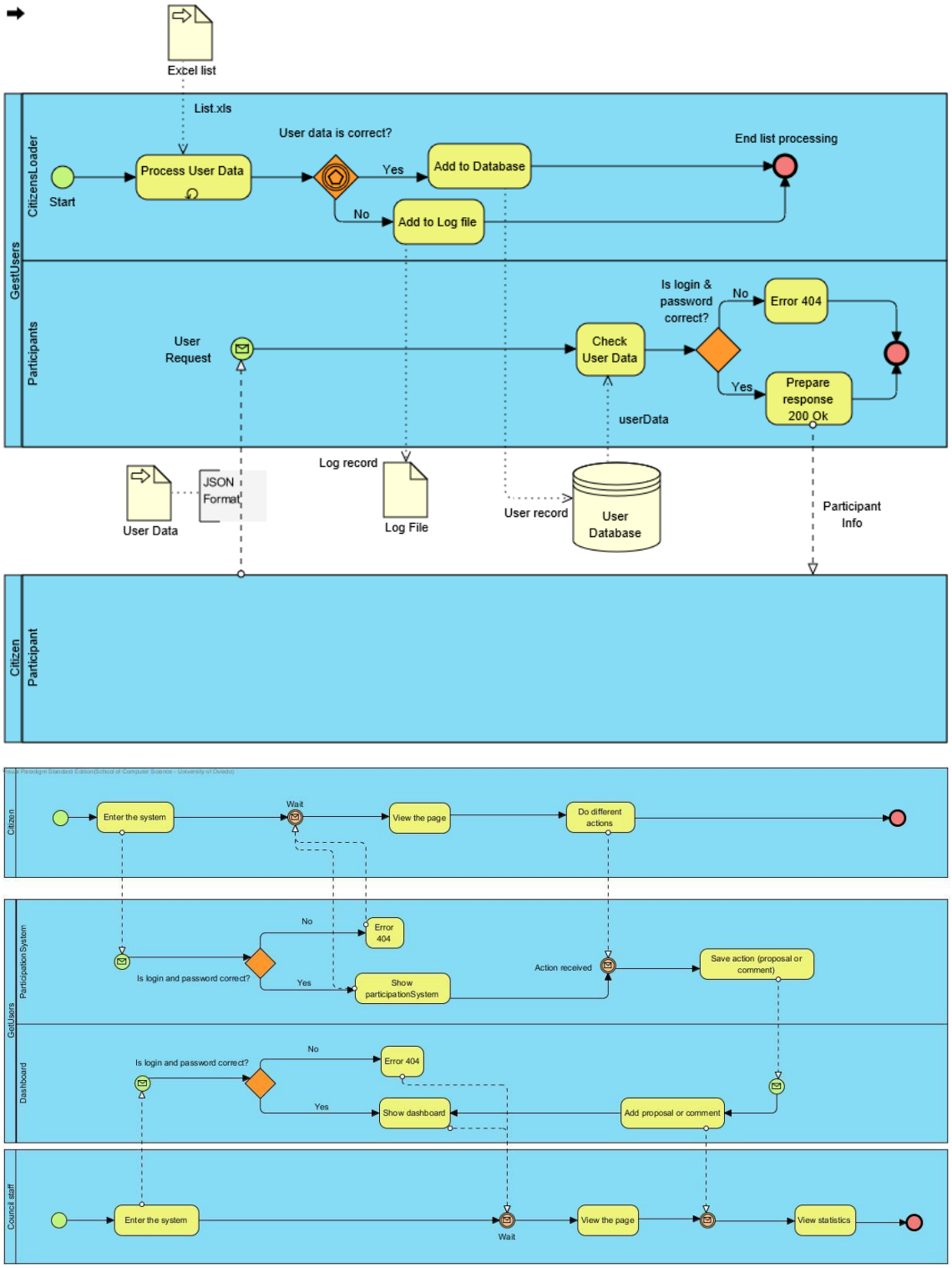

Figure 2. BPMN Diagram

8 Quality Scenarios

The table below contains the quality scenarios that have been identified:

Scenario	Source Stimulus	Stimulus	Environment	Artifact	Response	Measure	Affected Attribute Quality
1	Participation System	Ask information about a user	Runtime	Participants	Participation System obtains the required information in less than 15seg at any time in the day	The required information is obtained	**AT001**
2	Participation System	Ask information about a user	Runtime	Participants	Participation System obtains the required information for several users at the same time	The required information is obtained	**AT002**
3	Participation System	Ask information about a user	Runtime	Participants	Participation System obtains the results at the same time the information is required.	The requested results are obtained	**AT003**
4	Student developer	A new option is implemented for the report file	Development	ReportWriter, DBUpdate and Parser	The option is implemented with minimal changes that affect only the report writer module	Less than one day of work	**AT004**

Scenario	Source Stimulus	Stimulus	Environment	Artifact	Response	Measure	Affected Attribute Quality
5	Student developer	A new output format is added	Development	Participants and DBManagement	The new output format is included with minimal changes to existing code.	Less than one day of work .	**AT005**
6	Council staff	The option to configure the service is selected	Runtime	Participants	The options are selected	The selected options started to work as expected	**AT006**
7	Dashboard System	Changes must be done easily	Development	Dashboard	Restrict who and in what manner will see the dashboard	System has to provide modifiability	**AT007**
8	System administrator	Load an Excel file into the System (DB)	Runtime	Parser, DBUpdate and ReportWriter	Loading an excel file without errors is done in a reasonable time.	< 1 second for each 10 Participantsparticipant	**AT008**
9	System administrated	Receive new information	Runtime	Parser, DBUpdate and ReportWriter	Receive the information without errors	<1 second per each 10 participants	**AT009**
10	System administrated	Receive high amounts of information	Runtime	Parser, DBUpdate and ReportWriter	Receive high amounts of information	Process the received information without breaking	**AT010**
11	Dashboard System	Show the information in real time	Runtime	Dashboard	Show graphically the results of participation system	The graphs must be updated in real time	**AT011**
12	Student developer	Load an Excel file into the system (DB)	Development/ Runtime	Parser, DBUpdate and ReportWriter (Optional)	Loading data should be done in a safe way	It is not possible to get access to the users' personal data except by the system administrator who cannot get access to the password	**AT012**

Scenario	Source Stimulus	Stimulus	Environment	Artifact	Response	Measure	Affected Attribute Quality
13	Citizen	Privacy of votes	Runtime	Participation System	The votes must be anonymus	System has to provide privacy of the votes, don't show people the information	**AT013**
14	Participation System	Restrict the access to the system	Runtime security	Participation System	The access to the configuration must be controlled and only accesed by authorized people	The system must accept authorized people and deny not authorized ones	**AT014**
15	Dashboard System	Restrict who and in what manner will see the dashboard	Runtime security	Dashboard	The dashboard can only be seen by authorized people	The system must distinguish between authorized and not authorized users	**AT015**
16	System administrator	Loads an excel file into the DB	Runtime	Parser, DBUpdate and ReportWriter	The loading process is made in a reliable way and it is possible to check that the data has been loaded	There are no errors in the database, no repeated record, and no citizen has less information than expected	**AT016**
17	Participants	Get access to the application	Runtime	Participants	A user can get access to his data but not to other user's data	Access to data is enabled only if the pair user name/password is correct	**AT017**
18	Participation System and Dashboard	Test the system	Test phase	Dashboard	Prove that Kafka Stream shows the correct data	Tested with a simulator of random events	**AT018**
19	System administrator	Loads an excel file into the DB	Runtime	Parser, DBUpdate and ReportWriter	The loading process behaves in a usual way and the options available to run the system are easy to understand	The system shows help options if the user asks for them. The error messages and other information can be understood by technical people	**AT019**

Scenario	Source Stimulus	Stimulus	Environment	Artifact	Response	Measure	Affected Attribute Quality
20	Participation System	Use the system	Runtime	Participation System	The system behaves as expected and everything is easy and intuitive to use	The system shows all the options in an intuitive way	**AT020**
21	Dashboard System	Show statistics	Runtime	Dashboard	The system shows the results in a way impossible to misunderstand	The system shows the results in a simple way	**AT021**
22	Citizen Participation System	Access to the web service	Runtime	Participants	The participation System requests information about a user by passing a combination of user name and password	A 200 OK response is sent with the correct format if the combination is OK or a failure information is returned	**AT022**
23	Dashboard System, Participation System	Share data to show it	Runtime	Dashboard, Kafka	Good communication between both systems to show the statistics	Graphs must show the actual statistics when the comments and likes are done	**AT023**
24	Student developer	Develops the system	Development	Participants CitizensLoader	The student developers can implement the system	The system can be implemented and testes in 2/3 weeks by third year undergraduate students.	**AT024**
25	System administrator	Deploys the system	Deployment	CitizensLoader, Participants	The system is deployed in a production environment	The system can be deployed by a system administrator in less than an hour.	**AT025**
26	Participation System, Dashboard System	Manage flux of users	Runtime	Participation System, Dashboard	Same efficiency for 1-infinite users	The system design is not changed when the number of users grows exponentially	**AT026**

Table 6. List of quality scenarios

9 Views

In the following paragraphs the identified the views that will be documented following the learning guide instructions.

View	Stakeholders	Quality Attributes	Scenarios
Context	ST-01, ST-02, ST-03, ST-04, ST-05, ST-06	AT019, AT024, AT025	11, 19, 24, 25
CitizensLoader	ST-01, ST-02, ST-03, ST-04, ST-05	AT004, AT005, AT008, AT012, AT016, AT019, AT020, AT024, AT025	4, 5, 8, 12, 16, 19, 20, 24, 25
Participants	ST-01, ST-03, ST-04, ST-05	AT001, AT009, AT012, AT017, AT020, AT022, AT024, AT025	1, 9, 12, 17, 20, 22, 24, 25
ParticipationSystem	ST-01, ST-03, ST-04, ST-05, ST-06	AT001, AT002, AT006, AT009, AT010, AT011, AT012, T013, AT014, AT017, AT018, AT020, AT024, AT025, AT026	1, 2, 6, 9, 10, 11, 12, 13, 14, 17, 18, 20, 24, 25, 26
Dashboard	ST-01, ST-02, ST-04, ST-05, ST-06	AT003, AT007, AT011, AT012, AT013, AT015, AT018, AT020, AT021, AT023, AT024, AT025, AT026	3, 7, 11, 12, 13, 15, 18, 20, 21, 23, 24, 25, 26

9.1 Context

The System view is divided in four main sub-systems.

9.1.1 Main overview

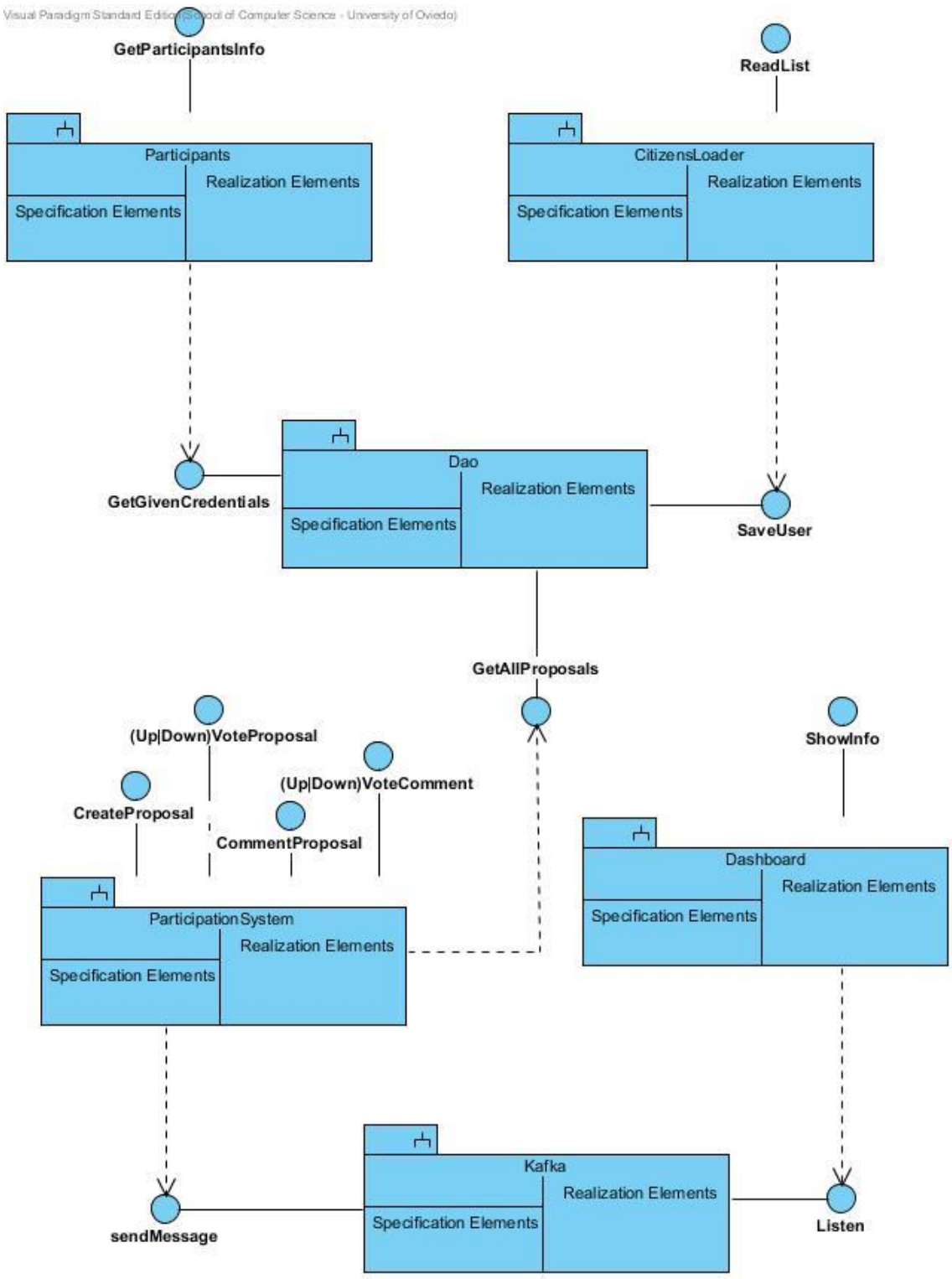

Figura 3. Context view

9.1.2 Elements Catalogue

9.1.2.1 Elements

Element	Properties
CitizensLoader	It introduces citizens data in the system. It reads an Excel file with data, generates passwords, personal letters and reports any errors.
Participants	This is the module used by citizens to check that their information is available in the system.
Dao	This module encapsulates database access.
ParticipationSystem	This module allows citizens to offer, comment and vote different proposals.
Dashboard	This is the module used to show the results of the system.
Kafka	This module encapsulates the use of Apache Kafka

9.1.2.2 Relationships

Citizens data are introduced in the system through the interface *ReadList* from module *CitizensLoader*. For each user, a password is generated as well as a personalized letter with information about the user.

That interface sends the data to the database through the interface *SaveUser* from the DataBase module.

The *Participants* module allows an external system to check the information about a user through the web service *GetParticipantInfo*. In order to check the information, *Participants* asks data to the *DataBase* module through the *GetGivenCredentials* interface.

The *Participation System* module allows users to access the application and use the whole system. This includes the possibility to make a proposal, comment other people's proposals, and voting both proposals and comments. The actions of add, comment or vote a proposal are done through CreateProposal, UpVoteProposal, DownVoteProposal, CommentProposal, UpVoteComment and DownVoteComment services.

The *Dasboard* module allows visualization of data from the system throw sendMessage interface. The data update is done by Kafka through *Listen* interface.

9.1.2.3 Interfaces/Ports

9.1.2.3.1 CitizensLoader

Interface	Type	Technology	Properties
ReadList	Interface	Command line invocation	This interface will be invoked from the main application as a console program

9.1.2.3.2 Participants

Interface	Tipo	Tecnología	Propiedades
GetParticipantInfo	Interface	Web Service	This interface will be invoked through an HTTP request

9.1.2.3.3 Dao

Interface	Tipo	Tecnología	Propiedades
getGivenCredentials	Interface	Method invocation	Returns data from citizens

saveUser	Interface	Method invocation	Inserts into the database data about a citizen included its password

9.1.2.3.4 ParticipationSystem

Interface	Tipo	Tecnología	Propiedades
CreateProposal	Interface	Method invocation	Creates a new proposal and sends the info to Kafka.
UpVoteProposal	Interface	Method invocation	Updates the proposal votes and sends a message to Kafka.
DownVoteProposal	Interface	Method invocation	Updates negatively the proposal votes and sents a message to Kafka.
CommentProposal	Interface	Method invocation	Creates a new comment and sends the info to Kafka.
UpVoteComment	Interface	Method invocation	Updates the comment votes and sends a message to Kafka.
DownVoteComment	Interface	Method invocation	Updates negatively the comment votes and sends a message to Kafka.
GetAllProposals	Interface	Method invocation	Obtains the info to show.
SendMessage	Interface	Method invocation	Sends the new info to kafka.

9.1.2.3.5 Dashboard

Interface	Tipo	Tecnología	Propiedades
ShowInfo	Interface	Method invocation	Shows the data from the system
Listen	Interface	Method invocation	Obtains the data to show

9.1.2.4 Behaviour

9.1.2.4.1 CitizensLoader

See 9.2.2.3.4.

It can also do the following options:

- The subsystem that generates the letters could implement the Adapter pattern which would enable to generate the letters in different formants in the future (Word, ODT, PDF, RTF, etc.).
- If the file contains errors, those errors should be detected and a report should be generated for its later treatment

9.1.2.4.2 Participants

It allows users to get access into the system to check if they can participate, using the information that they received in the letter. The users may not get access directly by a web browser, but through an external paticipation system that invokes the Participants module as a web service.

9.1.2.4.3 Dao

All the operations done in this module will be integrated in a *DAO pattern* which will contain the operations that offer access to the database. It encapsulates all the operations that affect the database.

9.1.2.4.4 ParticipationSystem

It allows citizens to create a proposal, as well as comment other proposals and vote them with the negative or positive option.

9.1.2.4.5 Dashboard

The dashboard will show the evolution of the participation system (proposals, comments and votes).

9.1.2.4.6 Kafka

The Kafka will be configurated to receive events from the participation system and send the data to the dashboard in order to update it.

9.2 Citizens Loader

9.2.1 Main overview

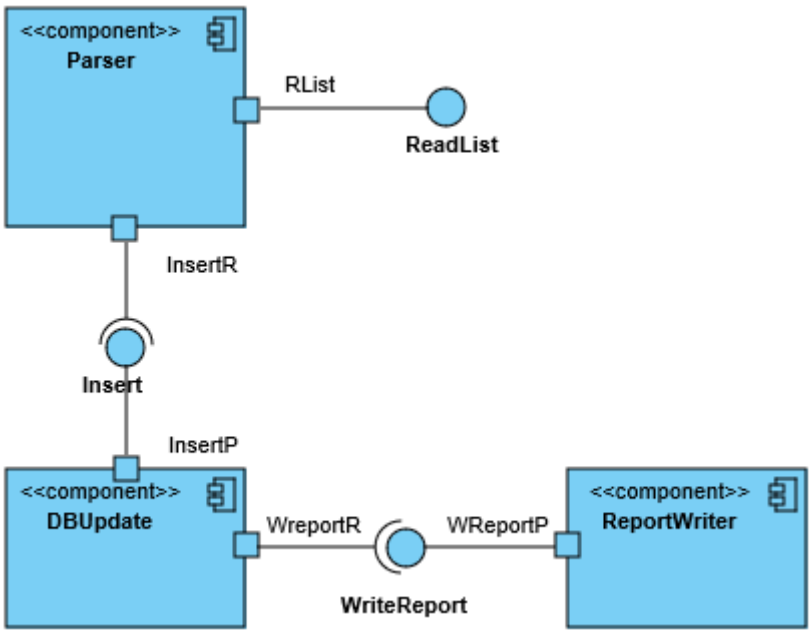

Figura 4. Citizens list view

9.2.2 Catalogue of Elements

9.2.2.1 Elements

Element	Properties
Parser	Reads data from the Excel file and transforms them into an in-memory object container that can be later iterated to insert the data in the

	database. It will also generate the *password* of the citizen as well as the personal letter. During the design and implementation this component can be divided into the sub-components needed to separate these services following the quality attributes AT004, AT005 and AT008.
DBUpdate	Encapsulates all the database operations using interfaces to allow the database access to be separated from some specific database implementations.
ReportWriter	It receives the pieces of data that were not possible to insert into the database as well as the reasons and writes a report containing all that information in a human-readable way

9.2.2.2 Relationships

The *Parser* component receives the input file in Excel format and reads and converts the information about the different users. It generates a new password for each user and adds the information to the database using the *DBUpdate* component.

If there are any errors during the loading phase (duplicated DNIs, empty DNI fields, etc.) or if the database component returns an error, this information will be notified to the Reportwriter component through the *WriteReport* interface.

9.2.2.3 Interfaces/ Ports

9.2.2.3.1 Parser

Interface	Type	Technology	Properties
ReadList	Interface	Method invocation	Read the Excel file with the citizens data.
Rlist	Port		Creates the needed subcomponents of the parser to process the input file.
Insert	Interface (Required)	Method invocation	It calls a method in the *DBUpdate* component to insert the information in the database.
InserR	Port		Verifies the data and creates the object to send to the *DBUpdate* component.

9.2.2.3.2 DBUpdate

Interface	Type	Technology	Properties
Insert	Interface	Method invocation	Receives and object with the information to insert in the database.
InsertP	Port		Verifies input data and generates and error if there is a lack of some mandatory attribute.
WriteReport	Interface (Required)	Method invocation	Calls a method from the *ReportWriter* component to write a new item in the report file.
WreportR	Port		Verifies the data to write

9.2.2.3.3 ReportWriter

Interface	Type	Technology	Properties
WriteReport	Interface	Method Invocation	Receives the data to write in the report file.
WreportP	Port		Adds data at the corresponding date and time.

9.2.2.3.4 Parser

Introduces the citizens data in the system obtained from Excel files that contain a row for each citizen. Each row (except the first one that contains the headings) contains the following columns:

- First name
- Last name
- Email
- Date of birth
- Address
- Nationality
- ID (National ID, the residence card ID, etc.)

Invocation will be done through a batch program executed in the command line by the system administrator. During the import process a password will be generated so the combination of email/password enable a user to enter the system and participate in the system receive information about the polling station code where the user can participate.

This component will also generate personal emails (letters) communicating the user that he has been added to the system with a user name (his email) and a password.

9.2.2.3.5 DBUpdate

It updates the database. See **¡Error! No se encuentra el origen de la referencia.**.

9.2.2.3.6 ReportWriter

It stores in a text file information about the errors that were produced by the conversion process. The basic information to store is:

- Date
- Time
- Original Excel file
- Error information (with all the needed information)

9.2.3 Context Diagram

See 9.1.

9.2.4 Rationale

The main design decisions of this sub-system are:

Scenario	Quality attributes	Justification
4	AT004	Defining an interface and an object for error reporting allows to add this functionality later.
5	AT005	Sending more than one letters each one in a different format allows to make sure that the receiver can read it.
8	AT008	Having a reasonale performance makes the user more confortable using the application.
12	AT012	Using a standard database which can be queried using SQL can allow the students to verify that the data has been correctly loaded.
16	AT016	The use of a batch application that can be executed manually or configured for its automatic execution is a common practice for system administrators.
19	AT019	The system administrator must know how to use the application.
20	AT020	The simple use of the application makes the user gets more confortable using

		it.
24	AT024	The web service API defined is simple and contains the minimal functionality. Leveraging on Spring Boot web framework will facilitate the development by the students given that the framework has solutions for all the required functionality
25	AT025	A batch application can be directly executed without any special needs for deployment

9.3 Participants

9.3.1 Main overview

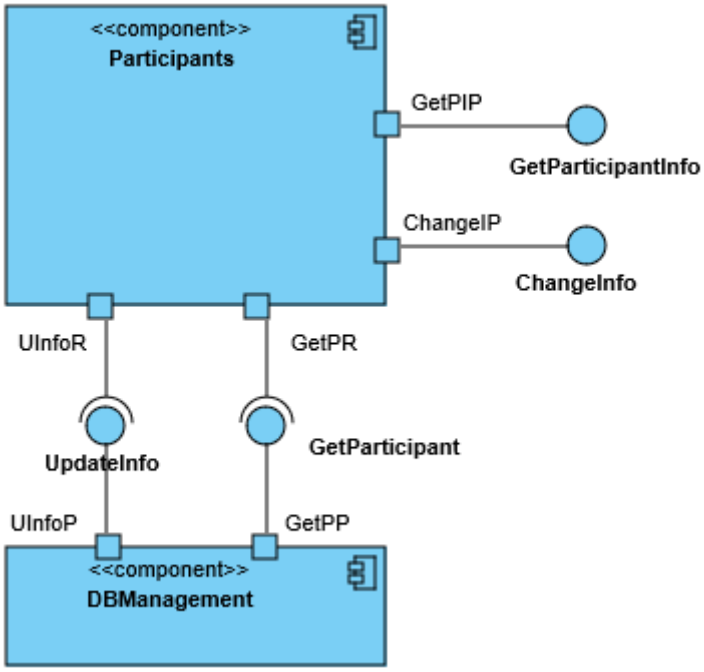

Figure 5. Participants View

9.3.2 Catalogue of elements

9.3.2.1 Elements

Element	Properties
Participants	It offers two web services: *GetParticipantInfo*, which allows to obtain information about a user and (Optional) *ChangePassword* that allows to change the password of a user (not implemented).
DBManagement	It offers two interfaces: GetParticipant, that returns the data of a participant from the database and (Optional) *UpdateInfo*, to update a password change in the database (not implemented).

9.3.2.2 Relationships

The ParticipantParticipation System invokes *Participants* using a web service call which is processed by *GetParticipantInfo* (sending *email/password*) and it gets access to the DBManagement system using the interface *GetParticipant*. If the email/password are correct the data is returned as a JSON response.

9.3.2.3 Interfaces/ports

9.3.2.3.1 Participants

Interface	Type	Technology	Properties
GetParticipantInfo	Interface	Web service	Allows to get access to a citizen data through the email/password combination
GetPIP	Port		Validates a user before asking the data.
ChangePassword	Interface	Web service	Allows to change a password using the combination: *email/password/newPasswod*.
ChangeInfo	Port		Validates a user before asking to change his password.
ChangeIP	Port		Validates a user before asking to change the password
UndateInfo	Interface (Required)	Method invocation	Asks a password change for a user.
UInfoR	Port		
GetParticipant	Interface (Requerida)	Method invocation	Asks information for the user
GetPR	Port		

9.3.2.3.2 DBManagement

Interface	Tipo	Tecnología	Propiedades
UndateInfo	Interface	Method invocation	Handles the password change of a user.
UInfoP	Port		
GetParticipant	Interface	Method invocation	Handles the information request for the user.
GetPP	Port		

9.3.2.4 Behaviour

9.3.2.4.1 Participants

It implements a REST web service to handle requests of information about users. The POST HTTP request will be done to the following address:

`<WebServiceURI>/user`

where `<WebServiceURI>` represents the URI where the web service has been deployed. The POST request contains JSON data with the following structure:

```
{"login": email, "password": password}
```

In case that the (email, password) combination are available in the database the response will be 200 OK with the a JSON body of the form:

```
{ "firstName": Nombre,
  "lastName": Apellidos,
  "age": Age (will be obtained from user's birth date and current time)
  "ID": User ID,
  "email": email
}
```

In case that the (email, password) is incorrect, the response will be `404 Not found`.

An HTML interface will be implemented so the web service can be used by humans through a web browser.

9.3.2.4.2 DBManagement

This component encapsulates all the database access so it can be easy to change the underlying database system.

9.3.3 Context Diagram

See 9.1.

9.3.4 Rationale

The main design decisions have been:

Scenario	Quality Attributes	Justification
1	AT001	Using a REST Web Service leverages on HTTP technology and makes it easier to deploy the system in some infrastructure with high availability.
9	AT009	Allows the user to be more confortable using the application.
12	AT012	Using a standard database which can be queried using SQL can allow the students to verify that the data has been correctly loaded.
17	AT017	The development of a REST web service based on JSON formats will facilitate the development of tests. The Spring Boot framework contains several tools for unit and integration testing of web applications that can be used. .
20	AT020	Allows the user to be more confortable using the application.
22	AT022	The use of a REST web service enables the automatic access to the system through a software client
24	AT024	The web service API defined is simple and contains the minimal functionality. Leveraging on Spring Boot web framework will facilitate the development by the students given that the framework has solutions for all the required functionality
25	AT025	A batch application can be directly executed without any special needs for deployment

9.4 ParticipationSystem

9.4.1 Main overview

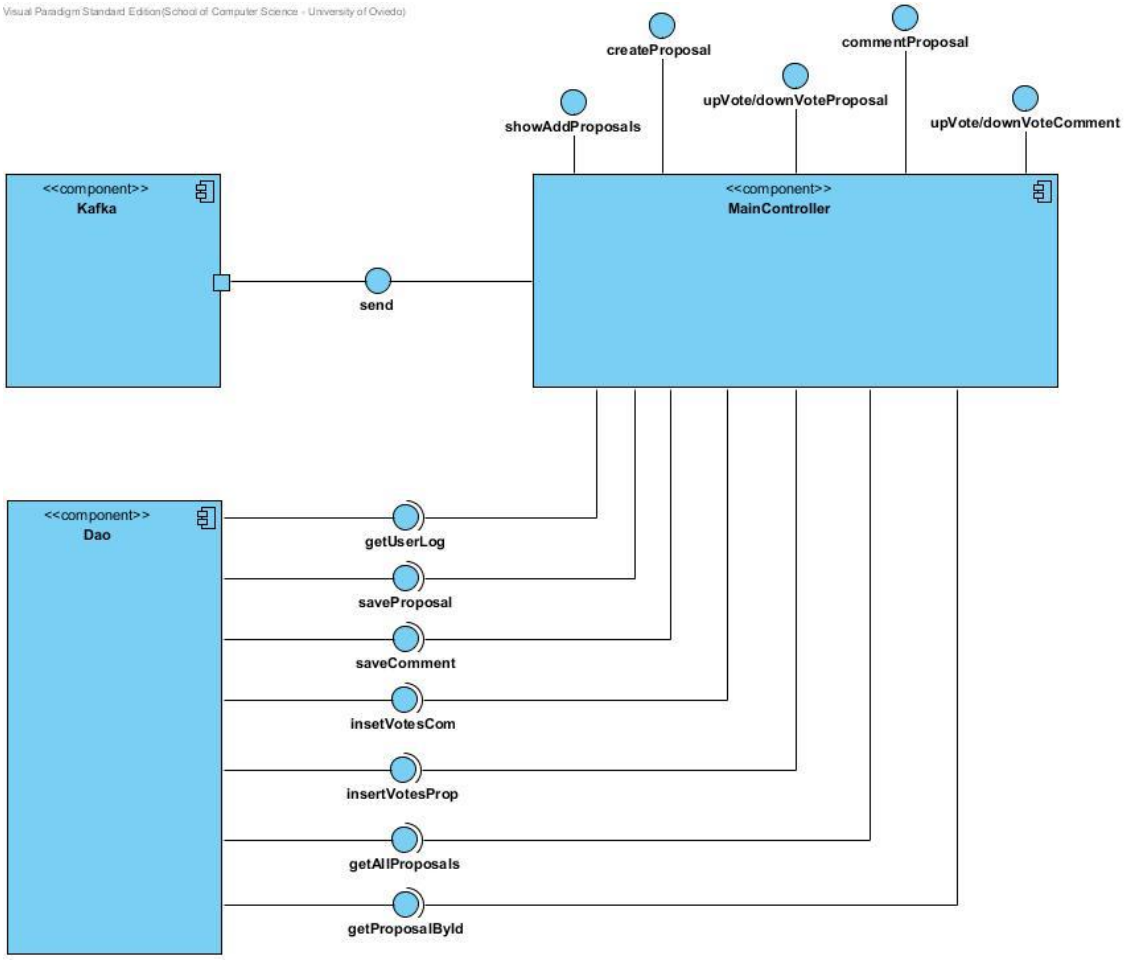

Figure 6. Participation System View

9.4.2 Catalogue of elements

9.4.2.1 Elements

Element	Properties
Kafka	Allows the ParticipationSystem to publish the streams of data needed for the Dashboard module.
MainController	Offers a webservice: *showAddProposals,* which allows the user to log in and it shows the user different information about the proposal and the comments and categories. It also offers interfaces: *createProposal,* which allows the user to create a new proposal; Up*VoteProposal,* which allows to vote a proposal positively; DowVoteProposal, which allows to vote a proposal negatively; *CommentProposal,* which allows the user to create a new comment on a proposal; Up*VoteComment,* which allows to vote a comment and DownVoteCommen, which allows to vote a comment in a

	negative way.
Dao	It offers four interfaces: *GetUserLog*, to handle the information request for the user; *saveProposal,* which creates a new proposal in the system; *setVotes,* which updates the votes; *saveComment,* which creates a new comment in the system.

9.4.2.2 Relationships

The MainController invokes showAddProposals, which also has a login. If the email/password are correct, the user is returned. The MainController then retrieves all the proposals in the system and offers the user to do one of the 4 actions: *CreateProposal, VoteProposal, CommentProposal*, and *VoteComment (both vote comment or proposal can be positive or negative).* Each of these invokations will do the appropriate update in the database as well as publish a new message in the kafka stream, which the corresponding topic needed.

9.4.2.3 Interfaces/ports

9.4.2.3.1 Kafka

Interface	Type	Technology	Properties
send	Interface	Method invocation	Allows to publish the logs of data in the streams published in Kafka's core

9.4.2.3.2 Dao

Interface	Type	Technology	Properties
getUserLog	Interface	Method invocation	Returns data from the user.
saveProposal	Interface	Method invocation	Inserts into the database a new proposal
saveComment	Interface	Method invocation	Inserts into the database a new comment of a proposal
insertVotesCom	Interface	Method invocation	Inserts into the database the votes of the comments
insertVotesProp	Interface	Method invocation	Inserts into the database the votes of the proposals.
getAllProposals	Interface	Method invocation	Allows to obtain all the existing proposals.
getProposalById	Interface	Method invocation	Allows to obtain a proposal given its id.

9.4.2.3.3 MainController

Interface	Type	Technology	Properties
showAddProposals	Interface	Web service	Allows a user access to the application through a combination of email/password
createProposal	Interface	Method invocation	Creates a new proposal and sends it to the database and the kafka stream
UpVoteProposal	Interface	Method invocation	Votes a proposal and sends it to the database and the kafka stream

DownVoteProposal	Interface	Method invocation	Votes a proposal negatively and sends it to the database and the kafka stream
commentProposal	Interface	Method invocation	Creates a new comment and sends it to the database and the kafka stream
UpVoteComment	Interface	Method invocation	Votes a comment and sends it to the database and the kafka stream
DownVoteComment	Interface	Method invocation	Votes a comment and sends it to the database and the kafka stream

9.4.2.4 Behaviour

9.4.2.4.1 Kafka

This component receives data from the MainController, and publishes it in the stream for all its subscribers.

9.4.2.4.2 Dao

This component encapsulates all the database access.

9.4.2.4.3 MainController

This component allows the user to create a new proposal, add comments to them and vote proposals and comments.

9.4.3 Context Diagram

See 9.1.

9.4.4 Rationale

The main design decisions have been:

Scenario	Quality Attributes	Justification
1	AT001	Using a REST Web Service leverages on HTTP technology and makes it easier to deploy the system in some infrastructure with high availability.
2	AT002	The system must serve millions of requests at the same time.
6	AT006	The system must allow different configuration options in order to select the appropriate ones in each case.
9	AT009	The system must be able to query information fast so the user doesn't wait too much time.
10	AT010	The system must process a big workload without breaking because it is thought to be used by lots of people simultaneously.
11	AT011	The system must show the changes at the same time they are being produced.
12	AT012	Using a standard database which can be queried using SQL can allow the students to verify that the data has been correctly loaded.
13	AT013	The system must ensure the confidentiality of votes.
14	AT014	The system must take into account that only allowed people have access to configure the system.
17	AT017	The development of a REST web service based on JSON formats will facilitate the development of tests. The Spring Boot framework contains several tools for unit and integration testing of web applications that can be used.
18	AT018	The system must test the Kafka results.
20	AT020	Using a simple system makes the user be more confortable with the system.
24	AT024	The web service API defined is simple and contains the minimal functionality. Leveraging on Spring Boot web framework will facilitate the development by the students given that the framework has solutions for all the required functionality

Scenario	Quality Attributes	Justification
25	AT025	A batch application can be directly executed without any special needs for deployment
26	AT026	The system must be prepared to support a high number of users.

9.5 Dashboard

9.5.1 Main overview

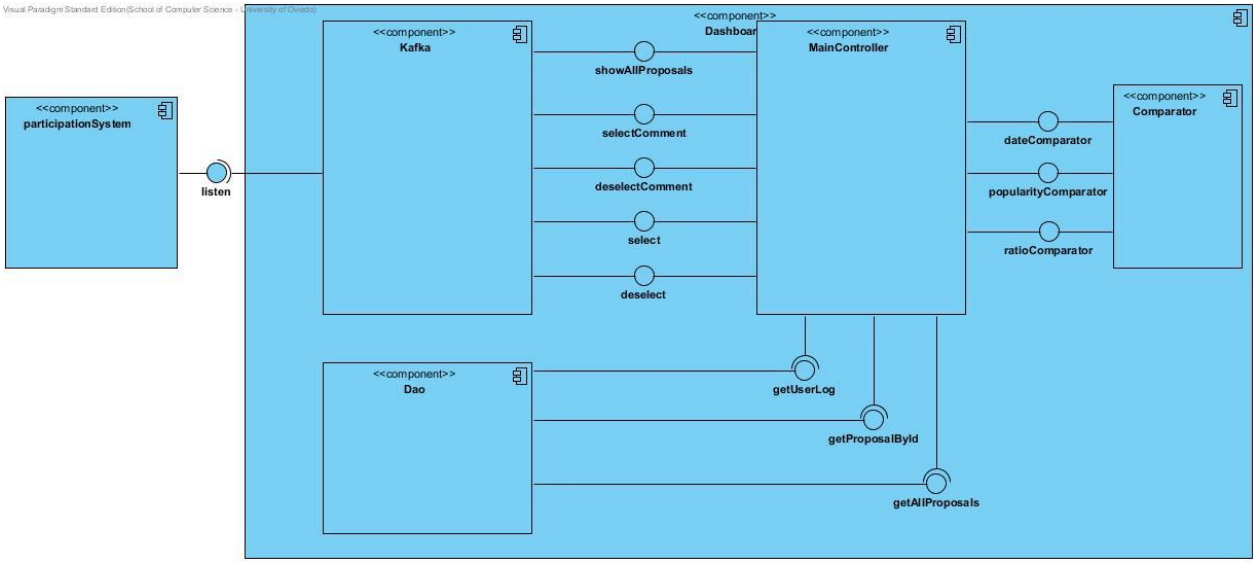

Figure 7. Dashboard View

9.5.2 Catalogue of elements

9.5.2.1 Elements

Element	Properties
Kafka	Provides the Subscribe interface, which is part of the Subscriber API given by Apache. This allows the Dashboard to receive the streams of data coming from the Participation System module.
Dao	Returns the information from the database.
MainController	Displays the information obtained.
Comparator	Allows to compare by different criteria.
ParticipationSystem	Provides the information to show

9.5.2.2 Relationships

The *Dashboard* system will obtain all the proposals from the *GetAllProposals* method invokation. Also, the Kafka component listens from the participationSystem. After that it deserializes such data and displays it in the screen.

9.5.2.3 Interfaces/ports

9.5.2.3.1 Kafka

Interface	Type	Technology	Properties
Listen	Interface	Method invocation	Allows to get the logs of data from the streams published in Kafka's core

9.5.2.3.2 Dao

Interface	Type	Technology	Properties
getProposalById	Interface	Method invocation	Obtains the proposal given an id
getAllProposals	Interface	Method invocation	Obtains all the proposals

9.5.2.3.3 Dashboard

Interface	Type	Technology	Properties
showAllProposals	Interface	Method invocation	Allows to get all the proposals
SelectComment	Interface	Method invocation	Select a comment showing its information
deselectComment	Interface	Method invocation	Deselect a selected comment
select	Interface	Method invocation	Allows to select
Deselect	Interface	Method invocation	Allows to deselect a selected thing.

9.5.2.3.4 Comparator

Interface	Type	Technology	Properties
dateComparator	Interface	Method invocation	Compares by date.
popularityComparator	Interface	Method invocation	Compares by popularity.
ratioComparator	Interface	Method invocation	Compares by ratio.

9.5.2.4 Behaviour

9.5.2.4.1 Kafka

This component receives a stream of data from the Participation System. After that, Kafka provides to the MainController a data stream containing all the records.

9.5.2.4.2 Dao

This component encapsulates all the database access.

9.5.2.4.3 MainController

This component gets the stream of data from Kafka, and then shows the data on the screen to the user user.

9.5.2.4.4 Comparator

This component allows to compare by different criteria.

9.5.2.4.5 ParticipationSystem

This component provides the stream of data to Kafka.

9.5.3 Context Diagram

See 9.1.

9.5.4 Rationale

The main design decisions have been:

Scenario	Quality Attributes	Justification
3	AT003	The system must show the results in real time.
7	AT007	The system must ensure that only authorized people have access to the dashboard.
11	AT011	The system must show the changes at the same time they are being produced.
12	AT012	Using a standard database which can be queried using SQL can allow the students to verify that the data has been correctly loaded.
13	AT013	The system must ensure the confidentiality of votes.
15	AT015	The system must allow access only for authorized people.
18	AT018	The system must test the Kafka results
20	AT020	Using a simple system makes the user be more confortable with the system.
21	AT021	The system must show the results in a clear way in order to not be misunderstood.
23	AT023	The system must show the results in real time without user interaction.
24	AT024	The web service API defined is simple and contains the minimal functionality. Leveraging on Spring Boot web framework will facilitate the development by the students given that the framework has solutions for all the required functionality
25	AT025	A batch application can be directly executed without any special needs for deployment
26	AT026	The system must be prepared to support a high number of users.

10 Package view and deployment view

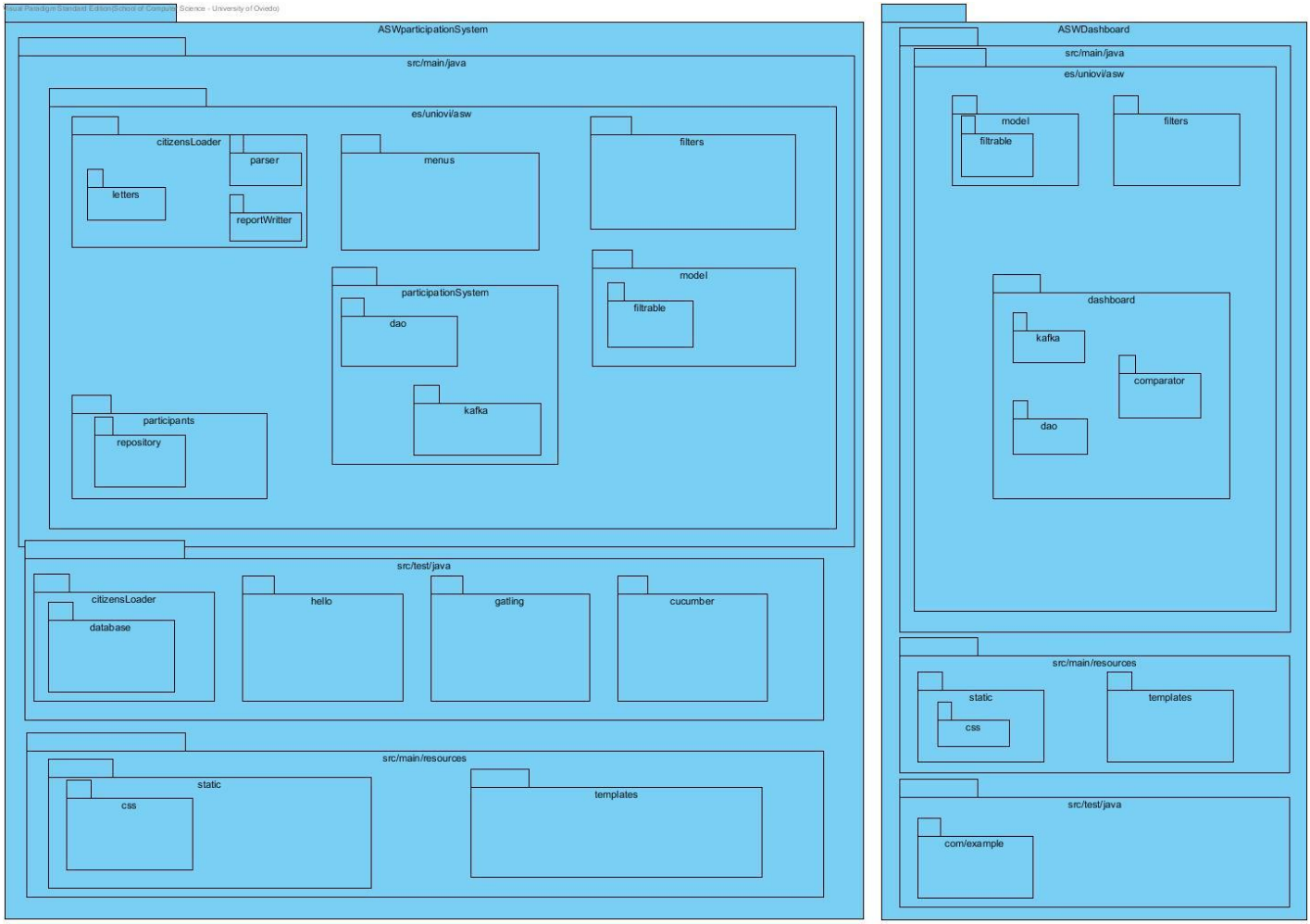

Figure 8. Package view

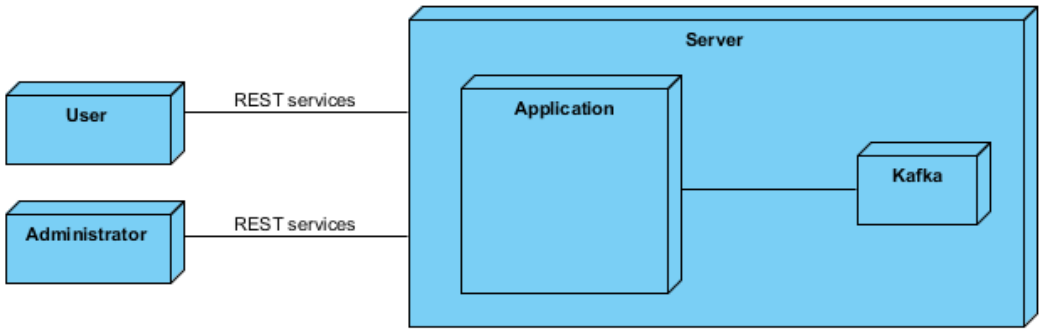

Figure 9. Deployment View

11 References

ANSI/IEEE 1471. (2000). *Recommended Practice for Architectural Description of Software-Intensive Systems.* ANSI/IEEE.

Bass, L., Clements, P., & Kazman, R. (2003). *Software Architecture in Practice, Second Edition.* Boston: Addison Wesley.