

Escuela de Ingeniería Informática, Univ. Oviedo

3 de mayo de 2016

GestUsers: Sistema de Gestión de Usuarios

Aquilino A. Juan Fuente

José Emilio Labra Gayo

Begoña Cristina Pelayo García-Bustelo

Jordán Pascual Espada

Vicente García Díaz

*Arquitectura Software para GestUsers. Descripción del trabajo práctico (2016)*

Descripción de la práctica a realizar por los grupos de trabajo de la asignatura de Arquitectura del Software durante el curso 2015-16.

**Grado de Ingeniería Informática del Software**

|  |  |
| --- | --- |
|  |  |

**GestUsers: Sistema de Gestión de Usuarios**

**Autores**: **D. Aquilino Adolfo Juan Fuente**

Doctor Ingeniero en Informática

Profesor del Departamento de Informática de la Universidad de Oviedo

**D. José Emilio Labra Gayo**

Doctor Ingeniero en Informática

Profesor del Departamento de Informática de la Universidad de Oviedo

**Da. Begoña Cristina Pelayo García-Bustelo**

Doctor Ingeniero en Informática

Profesora del Departamento de Informática de la Universidad de Oviedo

**Fecha**: 01 de enero de 2016

**Versión**: ￼

**Grado de Ingeniería Informática del Software**

|  |  |
| --- | --- |
|  |  |

**VoteSystem: Sistema de Votación**

**Autores**: **Carlos Villa Blanco**

Estudiante de ingeniería informática de la Universidad de Oviedo

**Jairo Montes Presa**

Estudiante de ingeniería informática de la Universidad de Oviedo

**Pablo Blanco Pacho**

Estudiante de ingeniería informática de la Universidad de Oviedo

**Fecha**: 08 de marzo de 2016

**Versión**: ￼

Tabla de contenido

[1 Introducción y Objetivos 6](#_Toc445760648)

[2 Requisitos 7](#_Toc445760649)

[2.1 Censuses 7](#_Toc445760650)

[2.2 VoterInfo 7](#_Toc445760651)

[2.3 VoteSystem 8](#_Toc445760652)

[2.4 AdminSystem 8](#_Toc445760653)

[3 Metodología usada 10](#_Toc445760654)

[4 Identificación de los Interesados) 11](#_Toc445760655)

[4.1 Alumnos que realizan la práctica 11](#_Toc445760656)

[4.2 Administrador del Sistema 11](#_Toc445760657)

[4.3 Votantes 12](#_Toc445760658)

[4.4 Desarrolladores del Sistema Electoral 12](#_Toc445760659)

[4.5 Profesores de la asignatura 12](#_Toc445760660)

[4.6 Desarrolladores del sistema de recuento y publicación 12](#_Toc445760661)

[4.7 La Junta Electoral 12](#_Toc445760662)

[4.8 La Mesa Electoral 12](#_Toc445760663)

[5 Atributos de calidad 13](#_Toc445760664)

[5.1 Lista de atributos de calidad 14](#_Toc445760665)

[5.2 Atributos de calidad￼ 15](#_Toc445760666)

[6 Restricciones 17](#_Toc445760667)

[6.1 Restricciones técnicas 17](#_Toc445760668)

[6.2 Restricciones organizativas 18](#_Toc445760669)

[7 Ámbito del sistema y contexto 19](#_Toc445760670)

[7.1 Sistema de gestión de usuarios 19](#_Toc445760671)

[7.2 Sistema de votación 20](#_Toc445760672)

[8 Escenarios de calidad 22](#_Toc445760673)

[9 Vistas 27](#_Toc445760674)

[9.1 Contexto 27](#_Toc445760675)

[9.1.1 Presentación principal 28](#_Toc445760676)

[9.1.2 Catálogo de elementos 28](#_Toc445760677)

[9.2 Census Reader 31](#_Toc445760678)

[9.2.1 Presentación principal 31](#_Toc445760679)

[9.2.2 Catálogo de elementos 31](#_Toc445760680)

[9.2.3 Diagrama contextual 33](#_Toc445760681)

[9.2.4 Justificación de las decisiones 33](#_Toc445760682)

[9.3 VoterInfo 34](#_Toc445760683)

[9.3.1 Presentación principal 34](#_Toc445760684)

[9.3.2 Catálogo de elementos 34](#_Toc445760685)

[9.3.3 Diagrama contextual 36](#_Toc445760686)

[9.3.4 Justificación de las decisiones 36](#_Toc445760687)

[9.4 AdminSystem 36](#_Toc445760688)

[9.4.1 Presentacion principal 36](#_Toc445760689)

[9.4.2 Catálogo de elementos 37](#_Toc445760690)

[9.4.3 Interfaces / Puertos 38](#_Toc445760691)

[9.4.4 Justificación de las decisiones 40](#_Toc445760692)

[9.5 VoteSystem 40](#_Toc445760693)

[9.5.1 Presentacion principal 40](#_Toc445760694)

[9.5.2 Catálogo de elementos 41](#_Toc445760695)

[9.5.3 Interfaces / Puertos 42](#_Toc445760696)

[9.5.4 Justificación de decisiones 43](#_Toc445760697)

[10 Bibliografía 44](#_Toc445760698)

# Introducción y Objetivos

El objetivo de este documento es definir la estructura de una arquitectura para gestión de usuarios que pueda ser reutilizada. Aunque el sistema aquí descrito tiene funcionalidad propia, el objetivo es que pueda integrarse como parte de un sistema general de voto electrónico.

La arquitectura aquí descrita forma parte del segundo entregable de la asignatura Arquitectura del Software, impartida en el grado de Ingeniería Informática del Software, Escuela de Ingeniería Informática, Universidad de Oviedo por los autores de este documento.

El sistema se ha descompuesto en tres partes: Censuses para cargar datos de un censo, VoterInfo, para consultar si un usuario puede votar, y un sistema para obtener los votos de los usuarios, el cual a su vez está subdividido en 2 partes: VoteSystem para permitir el voto de los usuarios, y AdminSystem, para configurar los parámetros de las elecciones e incorporar los votos obtenido en papel. Los estudiantes deberán implementar el software descrito en este documento en dos equipos de 3 o 4 personas durante 3 semanas. Un equipo desarrollará el módulo Censuses y el sistema para obtener los votos de los usuarios, y otro equipo desarrollará el módulo VoterInfo.

# Requisitos

La gestión de usuarios se hará siguiendo un esquema de dos módulos:

* Censuses: Carga de Censos
* VoterInfo: Comprueba que un usuario puede votar.

El sistema de votación se hará siguiendo un esquema de dos módulo:

* VoteSystem: Realización de la votación y almacenamiento de votos electrónicos.
* AdminSystem: Carga de las opciones de voto, lugares de votación y configuración del sistema, e incorporación de los votos obtenidos en papel.

## Censuses

El Administrador del sistema debe poder introducir los censos electorales. Dichos censos serán entregados por cada ayuntamiento con la lista de votantes y los datos del colegio electoral en el que deberán votar.

La Introducción de los censos se realizará a partir de ficheros Excel formados por filas de votantes, cada uno con la siguiente información:

* Nombre
* Email
* NIF
* Código de Colegio Electoral

Durante la importación del censo se creará un usuario y una clave aleatoria que le permita acceder al sistema para comprobar que está dado de alta, ver el colegio electoral en el que votará y durante el día de las elecciones, le permitirá emitir su voto.

Una vez importando el fichero Excel que contiene el censo, se emitirán cartas para cada usuario comunicándole que ha sido añadido al Censo Electoral, su usuario y su clave de acceso. Las cartas serán emitidas como ficheros de texto.

(Opcional) El sistema podría extenderse para emitir las cartas en otros formatos como Word o PDF.

(Opcional) Si el fichero viniera con errores, se detectarían y se enviarían los datos a un fichero de LOG para su posterior tratamiento.

(Opcional) El analizador de los datos de entrada debe ser configurable, ya que podrían venir los datos en diferentes formatos y no sólo en Excel. Es opcional permitir más de una entrada, pero es obligatorio que el sistema permita en el futuro una ampliación de manera sencilla.

(Opcional) Se puede extender el sistema para contemplar aspectos de seguridad.

## VoterInfo

Los usuarios deben poder acceder al sistema para comprobar que han sido dados de alta, a partir de la información de la carta recibida. Para ello se creará un servicio Web sencillo que tome como parámetros codificados en una llamada POST el nombre de usuario y la clave,y devuelva información sobre el código de colegio electoral del votante en caso correcto o informe del error en case incorrecto. Tanto los parámetros como la respuesta se enviarán en formato JSON.

(Opcional) Se creará un sencillo interfaz de acceso en HTML para que los usuarios puedan entrar en el sistema y consultar su información.

(Opcional) Mediante negociación de contenido, se podrá utilizar otros formatos como XML

(Opcional) Se puede extender la funcionalidad para que el servicio Web permita cambiar la clave de acceso al sistema.

(Opcional) Se puede extender el sistema para contemplar aspectos de seguridad.

## VoteSystem

Los votantes introducirán su usuario y contraseña en el sistema para poder realizar su votación, siempre y cuando accedan el día y a las horas establecidas por la Junta Electoral.

La mesa electoral introducirá (en el lugar habilitado para ello) su usuario y contraseña para acceder a los servicios, con los que podrá indicar que un usuario ya votó en el caso de que este lo haga físicamente. Este sistema también le permitirá saber si un usuario ya votó electrónicamente mostrándole un aviso en el caso de que esto haya ocurrido. Para realizar esta acción introducirá el DNI del usuario.

Una vez realizada la votación y terminado el recuento de los votos físicos, la mesa electoral tendrá que proporcionar a la Junta Electoral un fichero con el número de votos de cada partido. De modo que la Junta Electoral será la encargada de introducirlos en el sistema.

(Opcional) Se puede descomponer el sistema de votación en N-sistemas implantados de forma independiente (por ejemplo, un sistema por cada comunidad autónoma) y cuya información se sincroniza en el momento del recuento.

(Opcional) El sistema podría aceptar otros tipos de elecciones, como las elecciones al senado, al congreso, etc.

## AdminSystem

La Junta Electoral debe poder configurar los diferentes parámetros de un proceso electoral como son el día, la hora de inicio y fin de las elecciones, las opciones de voto o colegios electorales.

La introducción de los anteriores parámetros, además de los votos que se realizaron en papel, se realizará a partir de ficheros Excel.

A continuación se especifica la estructura de estos documentos:

1.Opciones de voto:

* Nombre

Se deben definir al menos 2 opciones de voto (sin contar el voto en blanco que se añade automáticamente).

2.Lugares de voto:

* Identificador
* Nombre
* Contraseña
* Ciudad
* País

3. Configuración:

* Fecha
* Hora inicio
* Hora fin

4.Votos:

* Nombre
* Nº Votos

# Metodología usada

Se va a realizar un estudio de arquitectura siguiendo el método de ADD￼￼￼(Bass, Clements, & Kazman, 2003)(Bass, Clements, & Kazman, 2003)￼(ANSI/IEEE 1471, 2000)￼

La documentación sigue el esquema propuesto en la guía de aprendizaje de la asignatura y también se han tomado algunas secciones siguiendo las plantillas propuestas en arc42 (<http://arc42.org/>). Las plantillas están tanto en español como en inglés y alemán.

Existe un proyecto que usa esas plantillas para documentar una arquitectura de software sencilla sobre una aplicación de gestión de rutas de bicicleta. La documentación está disponible en la Web. Se Pude ver aquí:

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

# Identificación de los Interesados

En este caso los interesados son:

1. Alumnos que realizan la práctica
2. Administrador del Sistema
3. Votantes
4. Desarrolladores del sistema electoral
5. Profesores de la asignatura
6. Desarrolladores del sistema de recuento y publicación
7. La Junta Electoral
8. La Mesa Electoral

Así pues, la lista de Stakeholders (interesados) queda:

| Código | Stakeholder | Intereses (Módulos) |
| --- | --- | --- |
| ST-01 | Alumnos que realizan la práctica | Todos |
| ST-02 | Administrador del Sistema | Carga de Censos |
| ST-03 | Votantes | Comprobaciones de los Votantes |
| ST-04 | Desarrolladores Sistema electoral | Comprobaciones de los Votantes |
| ST-05  ST-06 | Profesores de la asignatura  Desarrolladores del sistema de recuento y publicación | Todos  Realización de la votación |
| ST-07 | La Junta Electoral | Configuración del sistema de votación e inserción de votos físicos |
| ST-08 | La Mesa Electoral | Comprobación de votaciones |

Tabla 1. Lista de Stakeholders e intereses

Posteriormente se pasa a describir en más detalle cada uno.

## Alumnos que realizan la práctica

Se trata de los equipos de desarrollo.

Entre sus objetivos están:

* Utilizar tecnologías y metodologías conocidas, minimizando los riesgos relacionados con el aprendizaje de las nuevas.
* Aprender técnicas de desarrollo de software colaborativo y profesional.
* Utilización de tecnologías similares a las del grupo con quien deberán integrarse posteriormente para evitar incompatibilidades.

## Administrador del Sistema

Es la persona que carga los Censos.

Entre sus objetivos están:

* Tecnologías sencillas de los ficheros de entrada.
* Ficheros que puedan leerse por los humanos.
* Ser capaz de automatizar el proceso de carga de censos
* Ser capaz de depurar el proceso de carga en caso de errores

## Votantes

Son los usuarios finales del sistema.

Entre sus objetivos están:

* Sencillez de acceso a las opciones de voto.
* Ser capaz de votar desde su casa de una forma segura.
* Ser capaz de consultar el estado del sistema.
* Ser capaz de cambiar su información en el sistema, por ejemplo, la clave (Opcional)

## Desarrolladores del Sistema Electoral

Está formado por el equipo que desarrollará el Sistema Electoral. Entre sus objetivos están:

* Disponer de una forma sencilla de comprobar si un usuario puede o no votar
* Utilizar tecnologías fáciles de usar e interoperables con otros sistemas.

## Profesores de la asignatura

Son los responsables de los resultados de la práctica.

Entre sus objetivos están:

* Proponer tecnologías que ayuden a los estudiantes a adquirir habilidades relacionadas con Arquitectura del Software mediante el desarrollo de un proyecto práctico.
* Mostrar a los estudiantes un ejemplo de documentación de arquitectura.

## Desarrolladores del sistema de recuento y publicación

Son los responsable del recuento y publicación de los resultados de la votación.

## La Junta Electoral

Son los responsables de la gestión del proceso electoral.

Entre sus objetivos están:

* Tecnologías sencillas de los ficheros de entrada.
* Ficheros que puedan leerse por los humanos.
* Ser capaz de automatizar el proceso de entrada de los datos con las opciones de votos, lugares de votación, configuración, y, tras la votación, el número de votos recibidos en papel para cada opción de voto.
* Almacenar el número de votos físicos para cada partido en la base de datos.

## La Mesa Electoral

Son los responsables de indicar al sistema que un usuario ha votado físicamente. Cada mesa electoral dispondrá de una cuenta en una aplicación web donde llevarán a cabo esta acción.

# Atributos de calidad

Para el sistema se han identificado los siguientes atributos de calidad:

* **Disponibilidad**
  + Disponibilidad del sistema para procesar datos 24x7.
  + Acceso al sistema de votación solo el día y horas indicados
* **Modificabilidad**
  + Facilidad de cambio de diversas partes de la aplicación: Cambiar el parser de entrada de censos.
  + Facilidad de cambio de diversas partes de la aplicación: Añadir un fichero de log
  + Facilidad para modificar partes de la aplicación: otros formatos de salida para las cartas personales
  + Facilidad de cambio de diversas partes de la aplicación: Permitir el cambio de contraseña.
  + Facilidad para modificar partes de la aplicación: otros formatos mediante negociación de contenido
* Facilidad de cambio de diversas partes de la aplicación: Permitir la realización de diversos tipos de elecciones.
* Facilidad de cambio de diversas partes de la aplicación: Cambiar el parser de entrada de opciones de votos, lugares de votación y configuración.
* **Rendimiento**
  + El rendimiento del proceso de carga del censo es razonable
  + El rendimiento del proceso de carga de los datos de opciones de voto, lugares y configuración es razonable.
  + La consulta de información de un usuario por el servicio web debe ser rápida.
  + La inserción de datos al realizar la votación debe ser rápida
* **Seguridad**
  + Garantizar la confidencialidad de los censos.
  + Garantizar la confidencialidad de los votos.
  + Garantizar que una persona no puede votar más de una vez.
  + Garantizar que una persona ajena a la mesa electoral no puede acceder a los servicios que la mesa electoral puede utilizar.
* **Testabilidad**
  + Debe ser posible chequear automáticamente que los datos del censo se cargan adecuadamente
  + Debe ser posible chequear que el servicio Web se comporta de forma adecuada
* **Usabilidad**
  + El sistema de carga de datos debe poder ser usado por usuarios administradores de sistema familiarizados con herramientas tipo Unix.
  + Permite elegir destinatario de voto de entre una lista.
  + Garantizar el acceso simultáneo de usuarios al sistema de votos.
* **Interoperabilidad**
  + El presente sistema será usado por el Sistema de Voto Electrónico, el cual delegará en el sistema actual la gestión de usuarios. El subsistema VoterInfo deberá ser utilizado por un proceso automático para consultar el estado de los usuarios que quieran votar.
* **Simplicidad**
  + Los dos sub-sistemas deberán ser simples y fáciles de desarrollar
* **Fiabilidad**
  + El sistema de tener controlados los posibles fallos de persistencia.
* **Desplegabilidad**
  + El sistema debe ser fácilmente desplegable, especialmente en un servidor en la nube.

## Lista de atributos de calidad

| **Código** | **Descripción** | **Tipo de Atributo** | **Módulo afectado** |
| --- | --- | --- | --- |
| **AT001** | Disponibilidad del sistema para procesar datos 24x7 | Disponibilidad | VoterInfo |
| **AT002** | Facilidad de cambio de diversas partes de la aplicación: Cambiar el parser de entrada de censos. | Modificabilidad | Censuses |
| **AT003** | Facilidad de cambio de diversas partes de la aplicación: Añadir un fichero de log. | Modificabilidad | Censuses y AdminSystem |
| **AT004** | Facilidad para modificar partes de la aplicación:  Añadir otros formatos de salida para las cartas personalizadas | Modificabilidad | Censuses |
| **AT005** | Facilidad de cambio de diversas partes de la aplicación: Permitir el cambio de contraseña. | Modificabilidad | VoterInfo |
| **AT006** | Facilidad para cambiar partes de la aplicación: procesar y devolver información en otros formatos mediante negociación de contenido. | Modificabilidad | VoterInfo |
| **AT007** | El rendimiento del proceso de carga de datos en el censo es razonable (no demasiado lento, pero tampoco crítico) | Rendimiento | Censuses |
| **AT008** | El sistema debe garantizar la confidencialidad de los datos de los usuarios | Seguridad | Censuses y VoterInfo |
| **AT009** | Debe ser posible chequear que el servicio web se comporta adecuadamente | Testabilidad | VoterInfo y VoteSystem |
| **AT010** | Debe ser posible chequear el comportamiento del sistema de carga de datos | Testabilidad | Censuses y AdminSystem |
| **AT011** | El sistema debe poder ser usado por administradores de sistemas familiarizados con herramientas tipo Unix | Usabilidad | Censuses y AdminSystem |
| **AT012** | El servicio Web debe poder ser utilizado por procesos automáticos que consulten el estado de un usuario | Interoperabilidad | VoterInfo |
| **AT013** | El sistema debe ser sencillo y fácil de implementar | Simplicidad | Censuses, VoterInfo, VoteSystem y AdminSystem |
| **AT014** | El sistema debe ser fácilmente desplegable | Desplegabilidad | Censuses, VoterInfo y AdminSystem |
| **AT015** | El acceso al sistema solo es posible el día y horas indicados | Disponibilidad | VoteSystem y AdminSystem |
| **AT016** | Garantizar la confidencialidad de los votos | Seguridad | VoteSystem |
| **AT017** | La inserción de datos al realizar la votación debe realizarse en menos de 10 segundos | Rendimiento | VoteSystem |
| **AT018** | Garantizar que una persona no puede votar más de una vez | Seguridad | VoteSystem |
| **AT019** | Garantizar el correcto registro de los votos | Fiabilidad | VoteSystem |
| **AT020** | Garantizar el voto simultáneo de usuarios | Usabilidad | VoteSystem |
| **AT021** | Facilidad de cambio de diversas partes de la aplicación: Cambiar el parser de entrada de opciones de votos, lugares de votación y configuración. | Modificabilidad | AdminSystem |
| **AT022** | El rendimiento del proceso de carga de los datos de opciones de voto, lugares y configuración es razonable (no demasiado lento, pero tampoco crítico) | Rendimiento | AdminSystem |
| **AT023** | Garantizar que una persona ajena a la mesa electoral no puede acceder a los servicios que la mesa electoral puede utilizar. | Seguridad | VoteSystem |
| **AT024** | Permitir elegir destinatario de voto de entre una lista | Usabilidad | VoteSystem |
|  |  |  |  |
|  |  |  |  |

Tabla 2. Lista de atributos de calidad y tipos

## Atributos de calidad

Los diferentes atributos de calidad son de interés para alguno de los Stakeholders. La siguiente tabla muestra la lista de intereses para el proyecto actual:

| **Atributos**  **vs**  **Interesados** | **ST-01** | **ST-02** | **ST-03** | **ST-04** | **ST-05** | **ST-06** | **ST-07** | **ST-08** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **AT001** | X |  | X | X | X |  |  |  |
| **AT002** | X | X |  |  | X |  |  |  |
| **AT003** | X | X |  |  | X |  |  |  |
| **AT004** | X | X |  |  | X |  |  |  |
| **AT005** | X |  | X |  | X |  |  |  |
| **AT006** | X |  | X | X | X |  |  |  |
| **AT007** | X | X |  |  | X |  |  |  |
| **AT008** | X | X |  |  | X |  |  |  |
| **AT009** | X | X |  |  | X |  |  |  |
| **AT010** | X |  | X | X | X |  | x |  |
| **AT011** | X | X |  |  | X |  | x |  |
| **AT012** | X |  |  | X | X |  |  |  |
| **AT013** | X |  |  | X | X |  |  |  |
| **AT014** | X | X |  |  | X |  | X |  |
| **AT015** | X |  | X |  | X |  | X |  |
| **AT016** | X |  | X |  | X |  |  |  |
| **AT017** | X |  | X |  | X | X |  | X |
| **AT018** | X |  | X |  | X | X |  |  |
| **AT019** | X |  |  |  | X | X | X | X |
| **AT020** | X |  | X |  | X |  |  | X |
| **AT021** | X |  |  |  | X |  | X |  |
| **AT022** | X |  |  |  | X |  | X |  |
| **AT023** | X |  |  |  | X |  |  | X |
| **AT024** | X |  | X |  | X |  |  |  |

Tabla 3. Lista de intereses de los *stakeholders*

# Restricciones

Para realizar esta aplicación existen las siguientes restricciones

## Restricciones técnicas

| **Código** | **Restricción** | **Motivación** |
| --- | --- | --- |
| **TC001** | El lenguaje de programación será Java | Se asume que el equipo de desarrollo (ST001) tiene conocimientos de Java |
| **TC002** | Se utilizará una base de datos relacional para almacenar los datos | Se asume que el equipo de desarrollo (ST001) tiene conocimientos de bases de datos relacionales y existen múltiples librerías para trabajar con bases de datos relacionales desde Java |
| **TC003** | El servicio Web estará basado en estilo REST | El estilo REST es fácil de implementar y consumir. |
| **TC004** | Los datos de entrada vienen en formato Excel | Excel es un formato de datos bastante popular y existen varias librerías Java para procesar ficheros Excel |
| **TC005** | El formato de salida de las cartas personalizadas será texto plano | Con el fin de facilitar la implementación se propone generar cartas personalizadas mediante texto plano. El equipo de desarrollo puede opcionalmente implementar otros formatos |
| **TC007** | Pruebas automáticas y desarrollo basado en pruebas | Las pruebas deberán ser ejecutables automáticamente. Se propone un desarrollo basado en pruebas así como la utilización de técnicas de integración continua. |
| **TC008** | El servicio Web VoteInfo y VoteSystem se implementará mediante el *framework* Spring Boot | El framework Spring Boot se basa en Spring, que es un framework Java muy popular en la industria. Existen muchos ejemplos y material de ayuda para facilitar el aprendizaje por parte de los estudiantes. |
| **TC008** | El servicio Web VoteSystem se desarrollará utilizando una arquitectura MVC con el framework JSF y Primefaces | JSF es un framework para crear aplicaciones java J2EE basadas en el patron MVC. PrimeFaces es una extensión que se apoya en JSF, que nos proporciona una serie de utilidades y componentes. |
| **TC009** | El servicio Web VoteSystem será testeado con el entorno de pruebas de software Selenium | Selenium es un entorno de pruebas de software para aplicaciones basadas en la web, que incluye un lenguaje específico de dominio para pruebas (Selanese) para escribir pruebas en un amplio número de lenguajes de programación populares incluyendo Java. |

Tabla 4. Restricciones técnicas

## Restricciones organizativas

| **Código** | **Restricción** | **Motivación** |
| --- | --- | --- |
| **OC001** | Cada sub-sistema será implementado por un equipo pequeño de estudiantes. | El tamaño de los equipos será de unos 3 ó 4 estudiantes con el fin de que los estudiantes puedan aprender a desarrollar software de forma colaborativa mediante un proyecto simple. |
| **OC002** | La estructura de la base de datos será la misma para los 2 sub-sistemas | El pegamento entre los 2 sub-sistemas es la base de datos, cuya estructura debe ser acordada por los 2 equipos. |
| **OC003** | El código fuente será gestionado mediante el sistema control de versiones Git en un repositorio público en github | Los sistemas de control de versiones son utilizados por la mayoría de las empresas de desarrollo de software. Github ofrece un software de gestión de proyectos muy potente |

Tabla 5. Restricciones organizativas

# Ámbito del sistema y contexto

## Sistema de gestión de usuarios

Para describir la solución se utilizarán diagramas contextuales y texto.

La aplicación está partida en dos procesos:

* Censuses: Se encarga de la carga de los censos.
* Voters: Se encarga de las comprobaciones de los votantes.

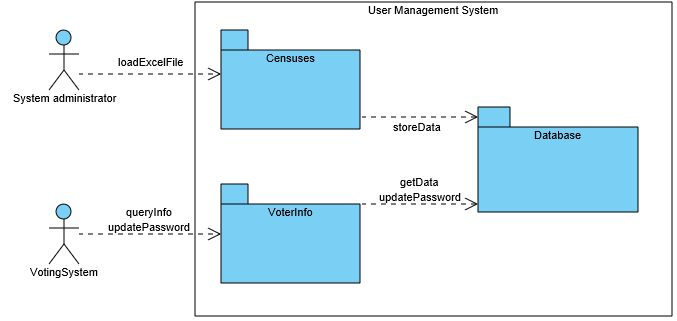


Figura 1. Contexto de negocio del sistema

A continuación se incluye un diagrama de flujo de datos del sistema:

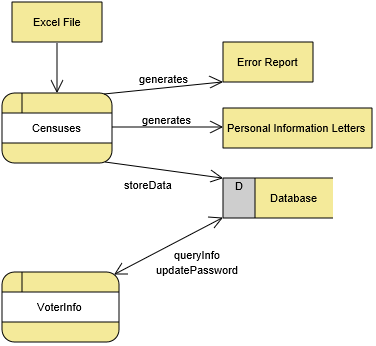


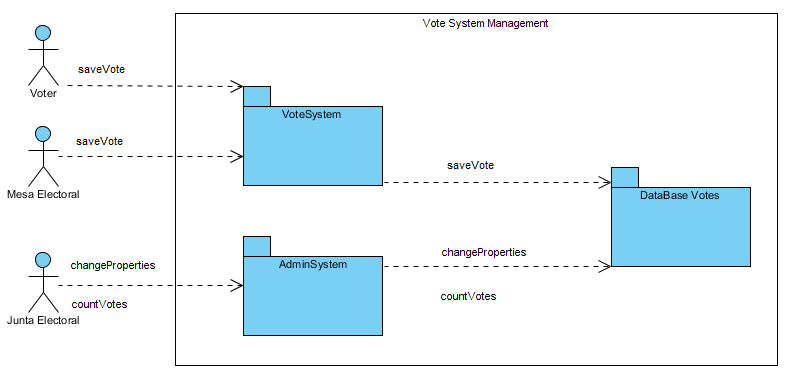
Figura 2. Diagrama de flujo de datos

## Sistema de votación

Para describir la solución se utilizarán diagramas contextuales y texto.

La aplicación está partida en dos procesos:

* Admin: Se encarga de la carga de los datos necesarios para realizar la votación (opciones de voto, lugares donde se realiza la votación y fecha y horas). Y tras la votación cargar los votos realizados en papel.
* Vote: Se encarga de recibir los votos e indicar que un usuario ya votó.

Figura 3.Contexto del negocio del sistema

A continuación se incluye un diagrama de flujo de datos del sistema:

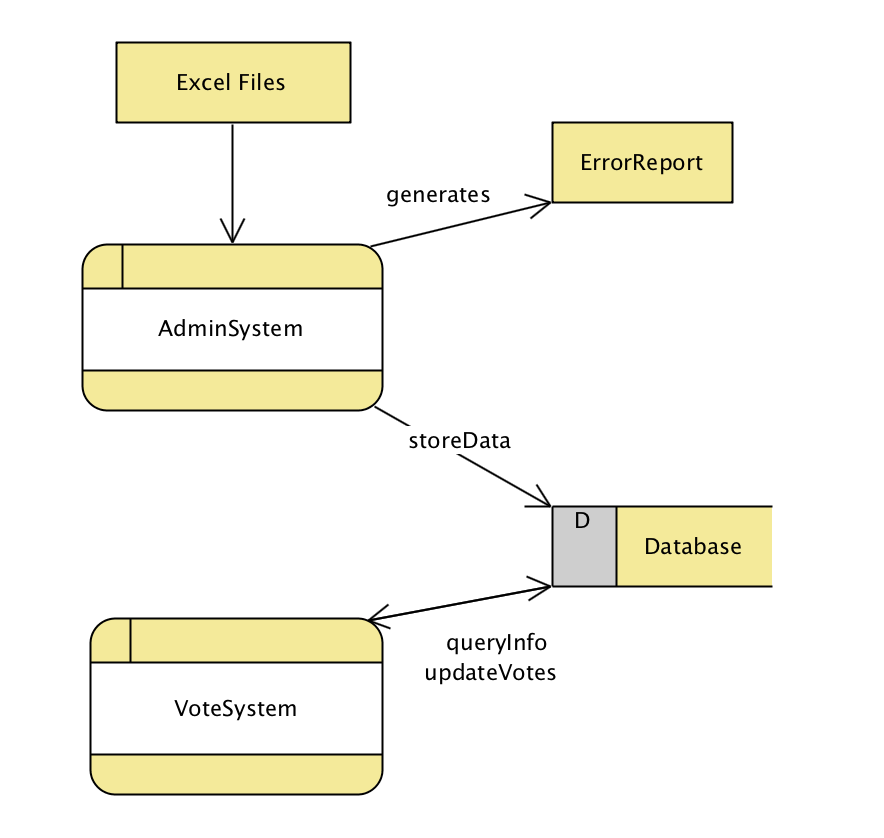


Figura 4. Diagrama de flujo de datos

# Escenarios de calidad

Con toda la información anterior se procederá a definir los escenarios de calidad que influencian esta arquitectura.

En las próximas páginas se muestra una tabla con la lista de escenarios identificados.

| Escenario Nº | Fuente de estímulo | Estímulo | Entorno | Artefacto | Respuesta | Medición de la respuesta | Atributo de calidad afectado |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Sistema de votación | Realiza una petición sobre un usuario | Tiempo de ejecución | VoterAccess | El sistema de votación recibe la respuesta adecuada | La información adecuada es recibida | **AT001** |
| 2 | Desarrollador | Se introduce un nuevo Parser | Desarrollo | Parser | La modificación es introducida adecuadamente | El sistema es compilado y pasa todas las pruebas | **AT002** |
| 3 | Desarrollador | Se implementa la generación de informes de error | Desarrollo | ReportWriter, DBUpdate Parser, DBVote y ConfParser | La opción es implementada con cambios mínimos que solamente afectan al módulo de generación de informes | Menos de un día de trabajo | **AT003** |
| 4 | Desarrollador | Un nuevo formato de salida es añadido | Desarrollo | VoterAccess y DBManagement | El nuevo formato de salida es añadido con los minimos camvbios. | Menos de un día de trabajo | **AT004** |
| 5 | Desarrollador | Opcion del cambio de la contraseña por parte del usuario. | Desarrollo | VoterAccess y DBManagement | La contraseña del usuario es modificada de forma correcta | Menos de un día de trabajo | **AT005** |
| 6 | Desarrollador | Un nuevo formato en añadido al servicio web | Desarrollo | VoterAccess | El nuevo formato es implementado | Menos de dos día de trabajo | **AT006** |
| 7 | Administrador del Sistema | Carga de un fichero excel en el sistema.(DB) | Tiempo de ejecución | Parser, DBUpdate y ReportWriter | Cargar un fichero excel sin errores en un tiempo razonable | < 1 segundo por cada 10 votantes | **AT007** |
| 8 | Desarrollador | Carga de un fichero excel en el sistema.(DB) | Desarrollo/  Tiempo de ejecución | Parser, DBUpdate y ReportWriter (Optional) | La carga de datos debe accerse de forma segura | No es posible acceder a los datos personales de los usuarios salvo el administrador del sistema, que tampoco puede acceder a las contraseñas. | **AT008** |
| 9 | Voters y Mesa electoral | Accede a la aplicación | Tiempo de ejecución | VoterAccess, UserVoting y PollingStationVoting | A user can get access to his data but not to other user's data | El acceso a los datos o al servicio de votación se permite solamente cuando la información de email/contraseña son correctas. | **AT009** |
| 10 | Administrador del Sistema | Carga un fichero Excel en la base de datos | Tiempo de ejecución | Parser, DBUpdate, ReportWriter, ConfParser, ReportWriterConf y DBVote | El proceso de carga se realiza de una forma fiable y es posible chequear que los datos han sido cargados adecuadamente. | No hay errores en la base de datos ni registros duplicados. Ningún votante tiene menos información que la requerida. | **AT010** |
| 11 | Administrador del Sistema | Carga un fichero Excel en la base de datos | Tiempo de ejecución | Parser, DBUpdate, ReportWriter, ConfParser, ReportWriterConf y DBVote | El proceso de carga se comporta de una forma habitual y las opciones son fáciles de comprender | El sistema muestra ayuda si el usuario la solicita. Los mensajes de error y otra información son comprensibles por personal técnico | **AT011** |
| 12 | Sistema de votación | Accede al servicio Web | Tiempo de ejecución | VoterAccess | El sistema de votación solicita información sobre un usuario pasando una combinación de email y contraseña | Se envía respuesta 200 OK si la combinación aparece en el sistema o error en caso contrario | **AT012** |
| 13 | Desarrollador | Implementa el sistema | Desarrollo | VoterInfo,  Censuses,  VoteSystem y  AdminSystem | Los desarrolladores pueden implementar el sistema | El sistema puede implementarse en 4 semanas | **AT013** |
| 14 | Administrador del Sistema | Despliega el sistema | Despliegue | Censuses, VoterInfo  Y AdminSystem | El sistema es desplegado en un entorno de producción | El sistema puede desplegarse en menos de una hora | **AT014** |
| 15 | Sistema de votación | Realiza una petición sobre un usuario | Tiempo de ejecución | VoteSystem y AdminSystem | El sistema de votación recibe la respuesta adecuada | La información adecuada es recibida | **AT015** |
| 16 | Sistema de votación | Voto a través del servicio web | Desarrollo | VoteSystem | El proceso de votación separa y no permite llegar a relacionar el voto y el votante | No es posible ver que opción de voto eligió un usuario | **AT016** |
| 17 | Sistema de votación | Voto a través del servicio web | Tiempo de ejecución | DBUpdate | La inserción de un voto en el sistema debe realizarse sin errores y en un tiempo razonable | < 10 segundos para insertar los datos | **AT017** |
| 18 | Sistema de votación | Voto a través del servicio web | Tiempo de ejecución | UserVoting, PollingStationVoting | El usuario no puede votar más de una vez | Muestra mensaje informando de que ya ha votado | **AT018** |
| 19 | Sistema de votación | Voto a través del servicio web | Tiempo de ejecución | VoteSystem | El proceso de votación se realiza correctamente. | No hay errores en la base de datos. Existe el mismo número de votantes que de votos. | **AT019** |
| 20 | Sistema de votación | Voto a través del servicio web | Tiempo de ejecución | VoteSystem | El proceso de votacion se produce sin errores en el sistema, ni cuelgues del mismo. | < 10 segundos para insertar el voto | **AT020** |
| 21 | Desarrollador | Se introduce un nuevo Parser | Desarrollo | ParserConf | La modificación es introducida adecuadamente | El sistema es compilado y pasa todas las pruebas | **AT021** |
| 22 | Junta Electoral | Cargar ficheros Excel en la base de datos | Tiempo de ejecución | ParserConf, ReportWriterConf y DBVote | Cargar los tres ficheros Excel sin errores se realiza en un tiempo razonable. | < 1 segundo por cada 10 filas | **AT022** |
| 23 | Desarrollador | Seguridad de los datos | Desarrollo | PollingStationVoting | Se deben marcar los usuarios que ya votaron en papel de manera segura | El sistema no debe permitir el acceso a los datos de una persona que no posea las claves | **AT023** |
|  |  |  |  |  |  |  |  |
| 24 | Sistema de votación | Voto a través del servicio web | Tiempo de ejecución | UserVoting | El sistema permite seleccionar de entre un listado de opciones de voto | < 5 segundos para mostrar el listado | **AT024** |
|  |  |  |  |  |  |  |  |

Tabla 6. Lista de escenarios de calidad

# Vistas

En los próximos párrafos se describirán algunas de las vistas identificadas y se documentarán de acuerdo a las instrucciones definidas en la guía de aprendizaje.

|  |  |  |  |
| --- | --- | --- | --- |
| Vista | Stakeholders | Atributos de calidad | Escenarios |
| Context | ST-01, ST-02, ST-03, ST-04, ST-05 | AT011, AT013 | 11, 13 |
| Censuses | ST-01, ST-02, ST-04, ST-05 | AT002, AT003, AT004, AT007, AT008 y AT010, AT011, AT013 | 2, 3, 4, 7, 8, 10, 11, 13 |
| VoterInfo | ST-01, ST-03, ST-04, ST-05 | AT001, AT005, AT006, AT008, AT009, AT012, AT013 | 1, 5, 6, 8, 9, 12, 13 |
| AdminSystem | ST-01, ST-05, ST-06, ST-07 | AT003, AT010, AT011, AT013, AT014, AT015, AT021, AT022 | 3, 10, 11, 13, 14, 15, 21, 22 |
| VoteSystem | ST-01, ST-03, ST-05, ST-06, ST-08 | AT009, AT013, AT015, AT016, AT017, AT018, AT019, AT020, AT023 ,AT024 | 9, 13, 15, 16, 17, 18, 19, 20, 24 |

En los catálogos y en las vistas se han definido tanto los elementos obligatorios como opcionales (voluntarios). Es responsabilidad de cada equipo eliminar los que no vaya a implementar.

## Contexto

La vista de sistema describe los dos subsistemas en interacción así como sus interfaces.

### Presentación principal

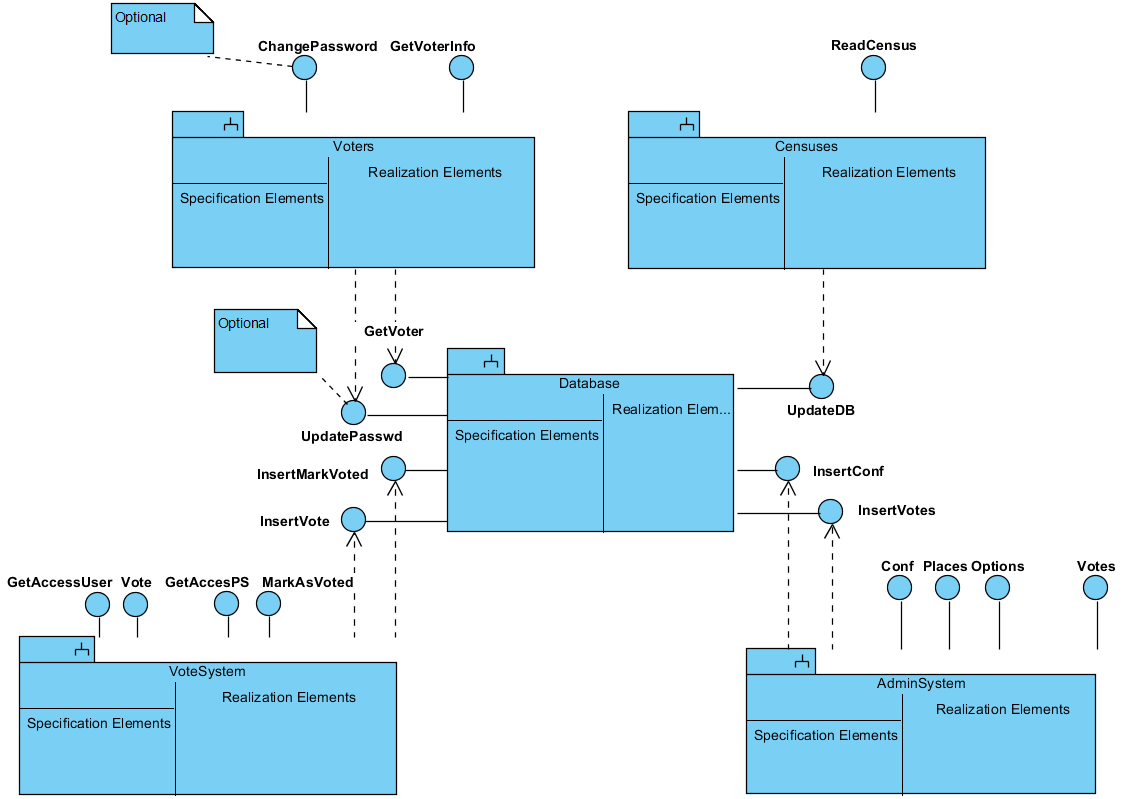


Figura 5. Context view

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| Censuses | Se encarga de la introducción de los censos en el sistema. Lee un fichero con los datos de los votantes. Crea las calves. Crea las cartas para los usuarios dados de alta. |
| Voters | Es el módulo usado por los votantes para comprobar que han sido dados de alta y opcionalmente para hace el cambio de clave. |
| DataBase | Este módulo encapsula los accesos a la base de datos. |
| VoteSystem | Es el modulo encargado de realizar permitir el acceso al servicio al usuario, y a la mesa electoral. También es el encargado de realizar la votación online como usuario, y de permitir a la mesa electoral marcar que un votante ha realizado el voto correctamente. |
| AdminSystem | Se encarga de cargar la configuración del sistema, a través de varios ficheros excel, y de enviar el recuento final a la base de datos |

#### Relaciones

Los datos de los votantes se introducen en el sistema a través de las interface *ReadCensus* del módulo *Censuses*. Para cada usuario, se crea una clave y emite una carta con todos los datos del usuario.

Este los envía a la base de datos a través de la interfaz *UpdateDB* del módulo DataBase.

El módulo *Voters* permite al usuario entrar en sesión a través del servicio web *GetVoterInfo* para comprobar sus datos. Para ello *Voters* pide los datos al módulo *DataBase* a través de la interfaz *GetVoter*.

Opcionalmente se puede implementar la interface *ChangePassword* que permite al usuario cambiar su clave, para ello el módulo *Voters* solicita al módulo *DataBase* el cambio de clave a través de la clase *UpdatePasswd*.

Los ficheros de la configuración son recibidos por el *AdminSystem*, a través de las interfaces *Conf*, *Places*, y *Options*. Los procesa, y los guarda en la base de datos.

El fichero con el resultado de las votaciones, se recibe a través de la interfaz *Votes* del modulo de A*dminSystem*.

El modulo *VoteSystem* es el encargado del acceso de los usuarios, o del responsable de la mesa electoral a través de las interfaces *GetAccessUser*, y *GetAccessPS* respectivamente. También es capaz de guardar un voto en la base de datos a través de *InsertVote*, si el usuario inicio sesión de manera electrónica. Si se realiza de manera física, seria la junta electoral, previo inicio de sesión en el sistema, la encargada de indicar que un usuario ha votado a través de *InsertMarkVoted*

#### Interfaces / Puertos

##### Censuses

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| ReadCensus | Interface | Invocación mediante línea de comandos | Se invocará a partir de la aplicación principal como un programa en consola |

##### Voters

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| GetVoterInfo | Interface | Servicio Web | Este servicio se invocará a través de una petición HTTP |

##### DataBase

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| GetVoter | Interface | Invocación a Método | Devuelve los datos de un votante. |
| UpdateDB | Interface | Invocación a Método | Inserta en la base de datos los datos de un votante, incluida su contraseña |
| UpdatePasswd | Interface | Invocación a Método | Actualiza la clave del usuario en la base de datos. |
| InsertVote | Interface | Invocación a Método | Introduce un voto en la base de datos, e indica que un usuario ha votado de manera electronica |
| InsertMarkVote | Interface | Invocación a Método | Indica que un usuario ha votado de manera fisica |
| InsertConf | Interface | Invocación a Método | Inserta los datos recibidos de la configuracion en la base de datos |
| InsertVote | Interface | Invocación a Método | Inserta el recuento de votos finales en la base |

##### VoteSystem

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| GetAccesUser | Interface | Invocación a métodos | Este servicio se invocará a través de una petición HTTP |
| GetAccessPS | Interface | Invocación a métodos | Este servicio se invocará a través de una petición HTTP |
| Vote | Interface | Invocación a métodos | Este servicio se invocará a través de una petición HTTP |
| MarkAsVoted | Interface | Invocación a métodos | Este servicio se invocará a través de una petición HTTP |

##### AdminSystem

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| Conf | Interface | Invocación mediante línea de comandos | Se invocará a partir de la aplicación principal como un programa en consola |
| Places | Interface | Invocación mediante línea de comandos | Se invocará a partir de la aplicación principal como un programa en consola |
| Options | Interface | Invocación mediante línea de comandos | Se invocará a partir de la aplicación principal como un programa en consola |
| Votes | Interface | Invocación mediante línea de comandos | Se invocará a partir de la aplicación principal como un programa en consola |

#### Comportamiento

##### Censuses

Ver el párrafo 9.2.2.3.4.

Además puede hacerlas siguientes opciones:

* (**Opcional**) el subsistema que cree las cartas debería implementar un patrón *Adapter* que permita cartas cartas en el futuro en diferentes formatos (Word, ODT, PDF, RTF, ETC.).
* (**Opcional**) Si el fichero viniera con errores, se detectarían y se enviarían los datos a un fichero de LOG para su posterior tratamiento.
* (**Opcional**) El *parser* de los datos de entrada debería ser configurable mediante un patrón *Adapter*, para permitir datos en diferentes formatos (Excel, TXT, etc.).

##### Voters

Permite a los usuarios poder acceder al sistema para comprobar que han sido dados de alta, usando la información recibida en la carta. Los usuarios podrían no acceder directamente mediante un navegador Web, sino a través de un sistema de voto externo que invoca el módulo como un servicio Web.

##### DataBase

Todas las operaciones que realiza este módulo estarán integradas dentro de un patrón *Facade* donde estarán las operaciones servidas para el acceso a base de datos. Encapsula todas las operaciones de base de datos así como la tecnología usar.

## Census Reader

La vista de *Census Reader* muestra el primer nivel de descripción de los componentes.

### Presentación principal

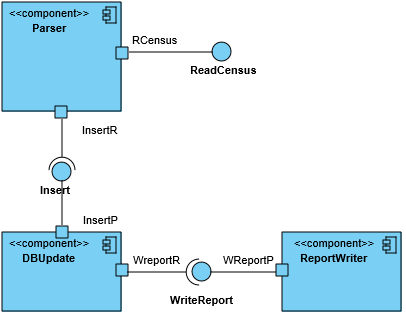


Figura 6. Vista Census Reader

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| Parser | Lee los datos de entrada en Excel y los transforma en un contenedor de objetos que pueda ser recorrido para su inserción en la base de datos.  También crea el *usuario/password* del votante y la carta de comunicación.  Durante el diseño y la implementación hay que partir este componente en los subcomponentes necesarios para separar todos estos servicios y hacerlo de manera que se cumplan los atributos de calidad AT002, AT003, AT004 y AT007. |
| DBUpdate | Encapsula todas las operaciones de base de datos usando interfaces para permitir el acceso a la base de datos. |
| ReportWritter | Recibe cadenas de información con los datos del usuario que fue imposible de dar de alta y las razones de dicho fallo y los escribe en un fichero de texto secuencial, indicando toda la información necesaria para poder revisar visualmente los fallos. |

#### Relaciones

El componente *Parser* recibe el fichero de entrada en *Excel* y mediante un *parser* convierte éste en objetos. Añade a éstos objetos el email y el *password*, y lo añade a la base de datos utilizando el componente *DBUpdate*.

(**Opcional**) Si se producen errores en la carga de datos (DNI duplicados, campo DNI vacío, etc.) o si el componente *de la base de datos* devuelve un error, esta información se escribe en un fichero de *LOG* mediante la interface *WriteReport* y el componente Report*Writer*.

#### Interfaces / Puertos

##### Parser

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| ReadCensus | Interface | Invocación a Métodos | Lee el fichero de *Excel* con los datos del censo. |
| Rcensus | Port |  | Crea los subcomponentes del *parser* necesarios para procesar el fichero de entrada. |
| Insert | Interface (Requerida) | Invocación a Métodos | Llama a un método del componente *DBUpdate* para hacer la inserción en la base de datos. |
| InserR | Port |  | Verifica los datos y crea el objeto a enviar a *DBUpdate*. |

##### DBUpdate

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| Insert | Interface | Invocación a Métodos | Recibe un objeto con la información para insertar en la base de datos. |
| InsertP | Port |  | Verifica los datos de entrada y si falta algún atributo obligatorio genera el correspondiente error. |
| WriteReport | Interface (Requerida) | Invocación a Métodos | Llama a un método del componente *ReportWriter* para escribir una línea en el fichero de *log*. |
| WreportR | Port |  | Verifica los datos a escribir. |

##### ReportWriter

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| WriteReport | Interface | Invocación a Métodos | Recibe los datos para escribir en el fichero de *log*. |
| WreportP | Port |  | Añade a los datos la hora y la fecha. |

##### Parser

Introduce los censos en el sistema a partir de ficheros Excel formados por filas de votantes, cada una con la siguiente información (excepto la primera fila que contiene las cabeceras):

* Nombre (String)
* Email (String con una formato acorde a las convenciones de correo electrónico)
* NIF (String formado por dígitos seguidos de una letra de verificación)
* Código de Colegio Electoral (nº entero)

La invocación se hará mediante un programa *batch* ejecutado en línea de comando por el administrador del sistema. Durante la importación del Censo se creará un usuario cuyo nombre de usuario coincidirá con el correo electrónico y se generará una contraseña aleatoria. La combinación adecuada de email/contraseña permitirá al usuario entrar al sistema, acceder a su información y emitir su voto.

Este componente también creará cartas personales comunicando al usuario que ha sido añadido al Censo Electoral, e informando de su correo electrónico y su clave de acceso.

##### DBUpdate

Actualiza la base de datos. Ver 9.1.2.4.3.

##### ReportWriter

(**Opcional**) Guarda en un fichero de texto la información de los errores producidos en el proceso deconversión. La información básica a guardar es:

* Fecha
* Hora
* Fichero Excel de procedencia
* Descripción del error (con toda la información necesaria)

### Diagrama contextual

Ver 9.1.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributos de calidad | Justificación |
| 2 | AT002 | El acceso al *parser* mediante un patrón *Adapter* garantiza un cambio rápido de *parser* sin tocar el código ya realizado en otras partes de la aplicación. |
| 3 | AT003 | Prever una interfaz y un objeto que pueda estar vacío para el informe de errores (*WriteReport*) facilita la modificabilidad en caso de añadir esta funcionalidad posteriormente. |
| 5 | AT005 | La utilización de una base de datos relacional ofrecerá un acceso eficiente a la información de los usuarios |
| 6 | AT006 | Utilizar una base de datos con características de seguridad habilitadas podrá garantizar que los datos están aislados de accesos indebidos. El envío de la contraseña por carta evita que la información pueda ser leída por medios electrónicos. |
| 8 | AT008 | La utilización de una base de datos relacional con acceso mediante SQL puede permitir a los alumnos verificar que los datos han sido cargados adecuadamente |
| 10 | AT010 | La utilización de una aplicación *batch* que pueda ser ejecutada manualmente o configurada para su ejecución automatizada es una práctica común entre los administradores de sistemas |
| 14 | AT014 | Una aplicación *batch* independiente puede ser ejecutada directamente sin ninguna necesidad especial para su despliegue |

## VoterInfo

### Presentación principal

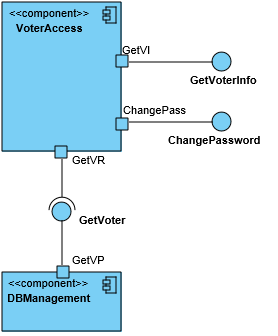


Figura 7.Vista de VoterInfo3

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| VoterAccess | Se accede a través de dos servicios web: *GetVoterInfo*, que permite al usuario acceder a sus datos en el sistema y (opcional) *ChangePassword* que permite al usuario cambiar su clave de acceso. |
| DBManagement | Se accede a través de dos interfaces: GetVoter, que devuelve los datos de un votante en la base de datos y (opcional) *UpdatePassword*, para actualizar un cambio de clave en la base de datos. |

#### Relaciones

El Sistema de Voto invoca *VoterAccess* utilizando una llamada a un servicio web que es procesada por *GetVoterInfo* (enviado *email/contraseña*) y éste accede a los datos encapsulados en DBManagement mediante la interface *GetVoter*. Si la combinación *email/contraseña* es correcta se devuelve la información del usuario en formato JSON.

(**Opcional**) El usuario (votante) accede a *VoterAccess* de forma manual al servicio web *ChangePassword* enviado *usuario/password/newPasswod*) y éste llama a la interface *UpdatePassword* para modificar la clave a través del componente *DBManagement*.

#### Interfaces / Puertos

##### VoterAccess

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| GetVoterInfo | Interface | Servicio Web | Permite el acceso a los datos de un votante mediante la combinación *email/contraseña*. |
| GetVI | Port |  | Hace la validación del usuario antes de pedir los datos. |
| ChangePassword | Interface | Servicio Web | Permite el cambio de clave a un votante mediante *email/contraseña/nuevaContraseña*. |
| ChangePass | Port |  | Hace la validación del usuario antes de solicitar el cambio de clave. |
| UpdatePassword | Interface (Requerida) | Invocación a Métodos | Solicita el cambio de clave para el usuario. |
| UpPssR | Port |  |  |
| GetVoter | Interface (Requerida) | Invocación a Métodos | Solicita la información para el usuario. |
| GetVR | Port |  |  |

##### DBManagement

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| UpdatePassword | Interface | Invocación a Métodos | Permite la solicitud de cambio de contraseña para el usuario. |
| UpPssR | Port |  |  |
| GetVoter | Interface | Invocación a Métodos | Permite la solicitud la de información para el usuario. |
| GetVP | Port |  |  |

#### Comportamiento

##### VoterAccess

Ver 9.3.2.2.

Implementa un servicio web REST para gestionar las peticiones de información sobre los usuarios. La petición principal será una petición HTTP POST que se realizará a la dirección:

<URIServicioWeb>/user

Donde URIServicioWeb representa la URI en la que está desplegado el servicio Web. La petición POST contiene datos JSON con la siguiente estructura:

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

En caso de que la combinación (email, password) aparezca en la base de datos, la respuesta será 200 OK con el cuerpo JSON de la forma:

{"name": name, "NIF": NIF, "email": email, "PollingStationCode": code}

En caso de que la combinación (email, password) no aparezca, la respuesta será 404 Not found.

(**Optional**) Se puede implementar un interfaz HTML para que el servicio Web pueda también ser utilizado por personas a través de un navegador Web convencional.

(**Optional**) El servicio Web puede ser extendido para permitir a los usuarios cambiar su password.

##### DBManagement

Encapsula todos los accesos a la base de datos.

### Diagrama contextual

Ver 9.1.

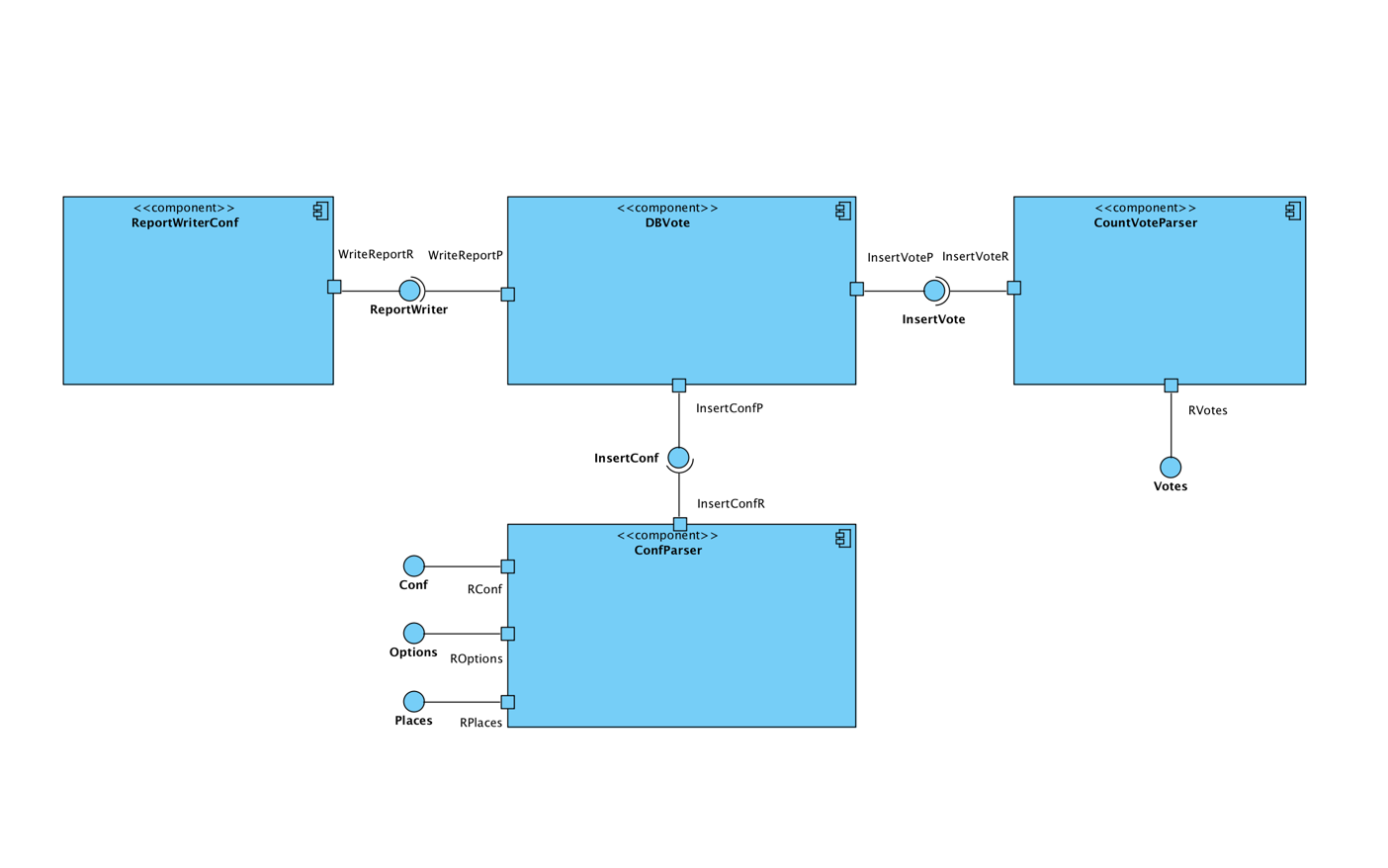
### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

| Escenario | Atributos de calidad | Justificación |
| --- | --- | --- |
| 1 | AT001 | La utilización de un servicio web REST se aprovecha de la tecnología HTTP y facilita el despliegue del sistema en infraestructuras de alta disponibilidad como pueden ser servidores Web, tanto locales como en la nube |
| 4 | AT005 | La encapsulación de las características del modelo que afectan a la base durante el desarrollo y la utilización de un framework basado en MVC facilitará el desarrollo de nuevas funcionalidades como las vistas basadas en HTML o el cambio de clabe de los usuarios. |
| 6 | AT006 | La utilización del framework Web Spring Boot facilitará el desarrollo de características comunes de la web como la negociación de contenido, dado que el framework ya contiene herramientas para su implementación. |
| 8 | AT008 | La restricción de acceso mediante *email/password* se considera suficientemente segura para este proceso. Las claves deberían almacenarse encriptadas. |
| 9 | AT009 | El desarrollo de un servicio web REST basado en formatos JSON facilitará la creación de pruebas. El framework Spring Boot contiene varias herramientas para pruebas unitarias y de integración. |
| 12 | AT012 | El uso de un servicio web REST permitirá el acceso automático al sistema a través de software cliente. |
| 13 | AT013 | El API del servicio web es simple y contiene la funcionalidad mínima necesaria. La utilización del framework Spring Boot facilitará el desarrollo por los estudiantes dado que el framework tiene soluciones para toda la funcionalidad requerida. |
| 14 | AT014 | La utilización del framework Spring Boot facilita el despliegue. Hay varios ejemplos que muestran cómo desplegar aplicaciones basadas en Spring Boot en servidores de producción. |

## AdminSystem

### Presentacion principal

Figura 8. Vista de AdminSystem

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| ConfParser | Lee los datos de entrada en Excel y los transforma en un contenedor de objetos que pueda ser creado todo el sistema de votación adaptado a los parámetros de entrada.  Durante el diseño y la implementación hay que tener en cuenta que se cargan varios parámetros distintos. |
| DBVote | Encapsula todas las operaciones de base de datos usando interfaces para permitir el acceso a la base de datos de votos. |
| ReportWriter | Recibe cadenas de información con los datos del usuario que fue imposible de dar de alta y las razones de dicho fallo y los escribe en un fichero de texto secuencial, indicando toda la información necesaria para poder revisar visualmente los fallos. |
| CounterVoteParser | Lee los datos de entrada en Excel y los transforma en un contenedor de objetos que pueda ser usado para guardar los votos físicos obtenidos en el recuento. |
|  |  |

**2.Relaciones**

El sistema de configuración invoca a los distintos métodos que cargan en el sistema todos los parámetros necesitados para la realización de una votación. Después de cargados todos los datos a través de la interfaz Insert lo datos son introducidos en DBVote.

(**Opcional**) Si se producen errores en la carga de datos (excel mal formado,campos vacios,etc.) o si el componente *de la base de datos* devuelve un error, esta información se escribe en un fichero de *LOG* mediante la interface *WriteReport* y el componente Report*Writer*.

### Interfaces / Puertos

##### ConfParser

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| InsertConf | Interface | Invocación a  métodos | Permite introducir los datos de la votación a la base de datos |
| Conf | Interface | Invocación  a métodos | Lee el fichero de configuración |
| Options | Interface | Invocación  a métodos | Lee el fichero de opciones |
| Places | Interface | Invocación  a métodos | Lee el fichero de lugares de foto |
| RConf | Port |  | Crea los subcomponentes del *parser* necesarios para procesar el fichero de configuración. |
| ROptions | Port |  | Crea los subcomponentes del *parser* necesarios para procesar el fichero de optiones de la votación. |
| RPlaces | Port |  | Crea los subcomponentes del *parser* necesarios para procesar el fichero de lugars de voto. |
| InsertConfR | Port |  | Verifica los datos y crea el objeto a enviar a DBVote. |
|  |  |  |  |

##### CountVoteParser

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| InsertConf | Interface | Invocación a  métodos | Permite introducir los datos de la votación a la base de datos |
| Votes | Interface | Invocación  a métodos | Lee el fichero de votos |
| RVotes | Port |  | Crea los subcomponentes del *parser* necesarios para procesar el fichero de votos. |
| InsertVoteR | Port |  | Verifica los datos y crea el objeto a enviar a DBVote. |
|  |  |  |  |

##### DBVote

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| Insert | Interface | Invocación  a metodos | Recibe un objeto con la información para introducir en la base de datos. |
| InsertConfP | Port |  | Verifica los datos de entrada y genera el error correspondiente. |
| WriteReport | Interface | Invocación a  Metodos. | Permite recoger los posibles errores. |
| WriteReportR | Port |  | Verifica los datos a escribir. |
| InsertVoteP | Port |  | Verifica los datos, y crea la lista de objetos para enviar a DBVote. |
| InsertVote | Interface | Invocacion  a metodos | Permite introducir la lista de votos en la base de datos. |
|  |  |  |  |

##### ReportWriter

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| WriteReport | Interface | Invocaión  a metodos | Recibe los datos para escribir en el Log. |
| WReportP | Port |  | Añade a los datos la hora y la fecha. |

##### ConfParser

Introduce los datos de los tres ficheros en el sistema a partir de ficheros Excel formados por filas, cada una con la siguiente información (excepto la primera fila que contiene las cabeceras):

-Fichero de configuración

* Fecha(Date)
* Hora inicio(Date)
* Hora fin(Date)

-Fichero de opciones

* Opciones de voto(String)

-Fichero de lugares(places)

* Nombre (String)
* Lugar(String)
* País(String)
* Código de Colegio Electoral (nº entero)

##### CountVoteParser

Introduce los datos del fichero Excel donde aparece recuento de los votos físicos.

-Fichero de votos(una columna con las opciones de voto y otra con el numero de votos)

La invocación se hará mediante un programa *batch* ejecutado en línea de comando por el administrador del sistema. Durante la importación del ficheros se creará un sistema de votación adaptado a las necesidades.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributos de calidad | Justificación |
| 2 | AT-021 | El acceso los distintos parser posibles mediante un patrón Strategy garantiza un cambio rápido de los paser sin tocar el código ya realizado en otras partes de la aplicación. |
| 3 | AT-003 | Prever una interfaz y un objeto que pueda estar vacío para el informe de errores (*WriteReport*) facilita la modificabilidad en caso de añadir esta funcionalidad posteriormente. |
| 5 | AT-022 | La utilización de una base de datos relacional ofrecerá un acceso eficiente a la información de los usuarios |
|  |  |  |
| 10 | AT-010 | La utilización de una aplicación *batch* que pueda ser ejecutada manualmente o configurada para su ejecución automatizada es una práctica común entre los administradores de sistemas |
| 14 | AT-014 | Una aplicación *batch* independiente puede ser ejecutada directamente sin ninguna necesidad especial para su despliegue |

## VoteSystem

### Presentacion principal

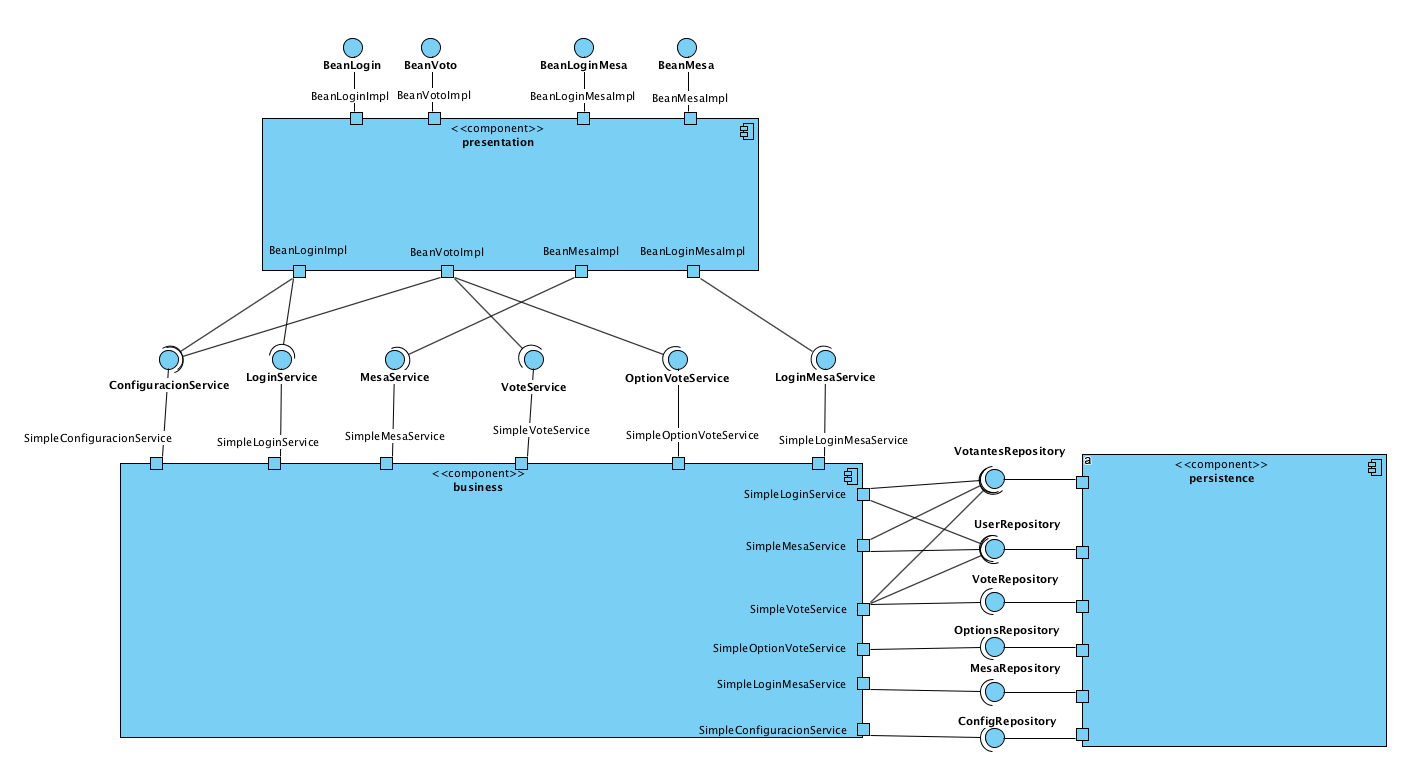


Figura 9. Vista de VoteSystem

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | *Propiedades* |
| presentation | Capa de presentación a la cual se accede utilizando 4 interfaces: BeanLogin utilizado por un votante para iniciar sesión, BeanVoto utilizado por el votante para votar, BeanLoginMesa usado por la mesa electoral para iniciar sesión y BeanMesa usado por la mesa electoral para marcar que un usuario ya votó. |
| business | Contiene la lógica del proyecto. |
| persistence | Encapsula todas las operaciones de base de datos usando interfaces para permitir el acceso a la base de datos de votos. |

#### Relaciones

El Sistema de Voto invoca *UserVoting* utilizando una llamada a un servicio web que es procesada por GetAccessUser (enviado *email/contraseña*).

El usuario (votante) accede a *VoterAccess* de forma manual al servicio web *Vote* enviado *voto* y éste llama a la interface *GetVoter* para reflejar el voto en DBUpdate.

El Sistema de Voto invoca *PollingStationVoting* utilizando una llamada a un servicio web que es procesada por GetAccessPs (enviado *email/contraseña*).

El usuario (presidente de la mesa) accede a *PollingStationVoting* de forma manual al servicio web *markAsVoted* enviado *la opción de votado* y éste llama a la interface *insertMarkVoted* para reflejar el voto en DBUpdate.

### Interfaces / Puertos

##### presentation

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| BeanLogin | Interface | Servicio Web | Permite el acceso a los datos de un votante mediante la combinación dni*l/contraseña*. |
| BeanMesaLogin | Interface | Servicio Web | Permite realizar el voto por parte de un usuario registrado. |
| BeanVoto | Interface |  |  |
| BeanMesa | Interface |  |  |
| BeanLoginImpl | Port |  | Acceso a la lógica para la comprobación del usuario. |
| BeanMesaLoginImpl | Port |  | Acceso a la lógica para la comprobación del usuario autorizado de la mesa. |
| BeanVotoImpl | Port |  | Acceso a la lógica para la realización del voto eléctronico. |
| BeasMesaImpl | Port |  | Acceso de la mesa electoral para la comprobación de voto de un usuario. |

##### business

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| LoginService | Interface | Servicio Web | Permite el acceso del usuario. |
| MesaService | Interface | Servicio Web | Permite al presidente de la mesa indicarle al sistema que un votando a votado de forma física. |
| VoteService | Interface |  | Permite el voto del usuario. |
| OptionVoteService | Interface |  | Permite obtener las opciones de voto que va tener el usuario. |
| LoginMesaService | Interface |  | Permite el acceso a la información necesaria por parte de la mesa electoral mediante la combinación dni*/contraseña*. |
| ConfiguracionService | Interface |  | Permite obtener los datos de configuración de la votación. |
| SimpleLoginService | Port |  | Hace la validación del usuario de la mesa electoral antes de pedir los datos. |
| SimpleMesaService | Port |  | Hace la validación de que se pueda realizar el marcado como votado de un usuario. |
| SimpleVoteService | Port |  | Comprueba la correcta selección de opciones para indicar que un usuario ha votado. |
| SimpleOptionVoteService | Port | Invocación a metodos | Permite invocar al método marcar usuario como que ha votado en la base de datos. |
| SimpleLoginMesaService | Port |  | Permite invocar al método que devuelve los datos de login de un usuario. |
| SimpleConfiguracionService | Port |  | Permite invocar al méodo que devuelve los datos de configuración de las elecciones. |

##### persistence

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| VotantesRepository | Interface |  | Verifica los datos de entrada para la realización de un voto. |
| UserRepository | Interface |  | Verifica los datos de entrada para marcar como votado a un usuario. |
| VoteRepository | Interface |  | Realiza la inserción del voto en la base de datos. |
| OptionRepository | Interface |  | Refleja que un usuario voto en la base de datos. |
| ConfigRepository | Interface |  | Refleja la configuración de la base de datos. |
| MesaRepository | Interface |  | Refleja la información obtenida por la mesa |

### Justificación de decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributos de calidad | Justificación |
| 8 | AT016 | La restricción de acceso mediante *email/password* se considera suficientemente segura para este proceso. Las claves deberían almacenarse encriptadas. |
| 13 | AT013 | El API del servicio web es simple y contiene la funcionalidad mínima necesaria. La utilización del framework PrimeFaces facilitará el desarrollo por los estudiantes dado que el framework tiene soluciones para toda la funcionalidad requerida y JSF para mostrar la información al usuario. |

# Bibliografía

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.