

Escuela de Ingeniería Informática, Univ. Oviedo

7 de marzo 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 primera 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**

|  |  |
| --- | --- |
| G:\Mis Documentos\Documentos\Universidad\Asignatures\Asig-Grado\Arquitectura del Software\Teoría\logo.png | **G:\Mis Documentos\Documentos\Universidad\Asignatures\Asig-Grado\Arquitectura del Software\Logos\logo-color.png** |

**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**:

Tabla de contenido

[1 Introducción y Objetivos 5](#_Toc442129668)

[2 Requisitos 6](#_Toc442129669)

[2.1 Censuses 6](#_Toc442129670)

[2.2 VoterInfo 6](#_Toc442129671)

[3 Metodología usada 8](#_Toc442129672)

[4 Identificación de los Interesados (Stakeholders) 9](#_Toc442129673)

[4.1 Alumnos que realizan la práctica 9](#_Toc442129674)

[4.2 Administrador del Sistema 9](#_Toc442129675)

[4.3 Votantes 9](#_Toc442129676)

[4.4 Desarrolladores del Sistema Electoral 10](#_Toc442129677)

[4.5 Profesores de la asignatura 10](#_Toc442129678)

[5 Atributos de calidad 11](#_Toc442129679)

[5.1 Lista de atributos de calidad 11](#_Toc442129680)

[5.2 Atributos de calidad e Interesados 12](#_Toc442129681)

[6 Restricciones 14](#_Toc442129682)

[6.1 Restricciones técnicas 14](#_Toc442129683)

[6.2 Restricciones organizativas 14](#_Toc442129684)

[7 Ámbito del sistema y contexto 15](#_Toc442129685)

[8 Escenarios de calidad 16](#_Toc442129686)

[9 Vistas 19](#_Toc442129687)

[9.1 Contexto 19](#_Toc442129688)

[9.1.1 Presentación principal 19](#_Toc442129689)

[9.1.2 Catálogo de elementos 19](#_Toc442129690)

[9.2 Census Reader 21](#_Toc442129691)

[9.2.1 Presentación principal 21](#_Toc442129692)

[9.2.2 Catálogo de elementos 21](#_Toc442129693)

[9.2.3 Diagrama contextual 23](#_Toc442129694)

[9.2.4 Justificación de las decisiones 23](#_Toc442129695)

[9.3 VoterInfo 24](#_Toc442129696)

[9.3.1 Presentación principal 24](#_Toc442129697)

[9.3.2 Catálogo de elementos 24](#_Toc442129698)

[9.3.3 Diagrama contextual 25](#_Toc442129699)

[9.3.4 Justificación de las decisiones 26](#_Toc442129700)

[10 Bibliografía 27](#_Toc442129701)

# 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 primer 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 dos partes: Censuses para cargar datos de un censo, y VoterInfo, para consultar si un usuario puede votar. Los estudiantes deberán implementar el software descrito en este documento en dos equipos de 3 ó 4 personas durante 3 semanas. Un equipo desarrollará el módulo Censuses y otro equipo desarrollará el módulo VoterInfo.

En los próximos entregables, los estudiantes crearán la arquitectura e implementarán un prototipo del resto del sistema de voto electoral del cual formará parte el sistema actual.

# Requisitos

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

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

## 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 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 un fichero Excel conteniendo un 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 ó 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.

## Sistema de recuento y publicación de votos

Una vez concluido el período de votaciones, el sistema deberá de realizar un recuento de los votos obtenidos por cada partido a partir de los datos de la BBDD. (No sé muy bien que habría que hacer con los colegios electorales en los que se vota a mano). El sistema deberá mostrar en tiempo real los resultados que se van obteniendo durante el escrutinio y al final del mismo mostrar los resultados y estadísticas finales

# Metodología usada

Se va a realizar un estudio de arquitectura siguiendo el método de ADD (Atribute-Driven Design) (Bass, Clements, & Kazman, 2003) y la norma del SEI (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 (Stakeholders)

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. Grupo que realiza la otra mitad de la práctica

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

| Código | Stakeholder | Intereses (Módulos) |
| --- | --- | --- |
| ST-01 | Alumnos que realizan la práctica | Ambos |
| 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  Grupo que realiza la otra mitad | Ambos  Sistema de recuento y publicación de votos |

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.
* Realizar un sistema en el que sea sencillo integrar un sistema de BBDD (implementado, se supone, por el otro grupo)

## 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 los datos.
* 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.

## Grupo que realiza la otra mitad del proyecto

Son los responsables de implementar el sistema de votación.

Entre sus objetivos están:

* Utilizar tecnologías y metodologías conocidas, minimizando los riesgos relacionados con el aprendizaje de las nuevas.
* Utilizar técnicas similares a las de nuestro grupo para la posterior integración de ambos proyectos.

# Atributos de calidad

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

* **Disponibilidad**
  + Disponibilidad del sistema para procesar datos 24x7.
* **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 para modificar las opciones de votación para poder cambiar el formato de elecciones según sean generales, un referéndum, etc.
* **Rendimiento**
  + El rendimiento del proceso de carga del censo es razonable
  + La consulta de información de un usuario por el servicio web debe ser rápida
  + El resultado producido durante el escrutinio deberá estar disponible en tiempo real.
* **Seguridad**
  + Garantizar la confidencialidad de los censos.
  + Garantizar la imposibilidad de acceso al sistema durante el proceso de almacenamiento o recuento de datos, para evitar una modificación ilegal en los mismos.
* **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.
* **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
  + Tanto el sistema de votación como el de recuento y publicación deben ser fácilmente integrables entre sí.
* + 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 |
| **AT004** | Facilidad para modificar oartes 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** | Facilidad para modificar las opciones de votación | Modificabilidad | Sistema de votación |
| **AT008** | El rendimiento del proceso de carga de datos en el censo es razonable (no demasiado lento, pero tampoco crítico. | Rendimiento | Censuses |
| **AT009** | El resultado producido durante el escrutinio deberá estar disponible en tiempo real. | Rendimiento | Sistema de recuento y publicación |
| **AT010** | El sistema debe garantizar la confidencialidad de los datos de los usuarios | Seguridad | Censuses y VoterInfo |
| **AT011** | El sistema debe garantizar que es imposible que un usuario pueda modificar los resultados almacenados en la BBDD. | Seguridad | Sistema de votación y de recuento y publicación. |
| **AT012** | Debe ser posible chequear que el servicio web se comporta adecuadamente | Testabilidad | VoterInfo |
| **AT013** | Debe ser posible chequear el comportamiento del sistema de carga de datos | Testabilidad | Censuses |
| **AT014** | El sistema debe poder ser usado por administradores de sistemas familiarizados con herramientas tipo Unix | Usabilidad | Censuses |
| **AT015** | El servicio Web debe poder ser utilizado por procesos automáticos que consulten el estado de un usuario | Interoperabilidad | VoterInfo |
| **AT016** | El sistema debe ser sencillo y fácil de implementar | Simplicidad | Censuses y VoterInfo |
| **AT017** | Ambos sistemas deben ser fácilmente integrables entre sí | Simplicidad | Sistema de votación y de recuento y publicación |
| **AT018** | El sistema debe ser fácilmente desplegable | Desplegabilidad | Censuses y VoterInfo |

Tabla 2. Lista de atributos de calidad y tipos

## Atributos de calidad e Interesados

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** |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |
| **AT008** | X | X |  |  | X |  |
| **AT009** | X |  |  |  |  |  |
| **AT010** | X | X |  |  | X |  |
| **AT011** | X |  |  |  |  | X |
| **AT012** | X | X |  |  | X |  |
| **AT013** | X |  | X | X | X |  |
| **AT014** | X | X |  |  | X |  |
| **AT015** | X |  |  | X | X |  |
| **AT016** | X |  |  | X | X |  |
| **AT017** | X |  |  |  |  | X |
| **AT018** | 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 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. |

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

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.

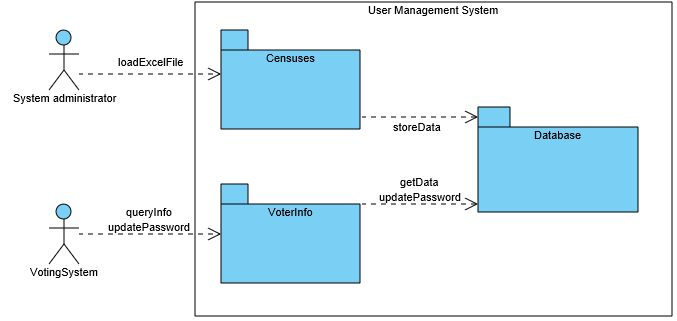


Figura 1. Contexto de negocio del sistema

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

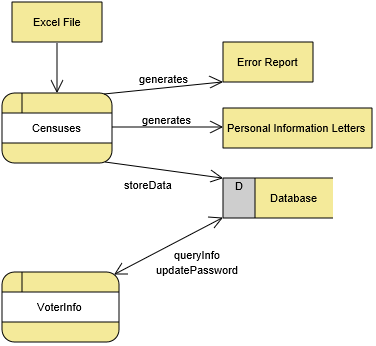


Figura 2. Diagrama de flujo de datos

# 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 y Parser | 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 | A new output format is added | Desarrollo | VoterAccess and DBManagement | The new output format is included with minimal changes to existing code. | Menos de un día de trabajo | **AT004** |
| 5 | Desarrollador | The option to change user's password is introduced | Desarrollo | VoterAccess and DBManagement | The password of a user is successfully changes | Menos de un día de trabajo | **AT005** |
| 6 | Desarrollador | A new format is added to the web service | Desarrollo | VoterAccess | The new format is implemented | Menos de dos día de trabajo | **AT006** |
| 7 | Administrador del Sistema | Load an Excel file into the System (DB) | Tiempo de ejecución | Parser, DBUpdate and ReportWriter | Loading an excel file without errors is done in a reasonable time. | < 1 segundo por cada 10 votantes | **AT007** |
| 8 | Desarrollador | Load an Excel file into the system (DB) | Desarrollo/  Tiempo de ejecución | Parser, DBUpdate and ReportWriter (Optional) | Loading data should be done in a safe way | 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 | Accede a la aplicación | Tiempo de ejecución | VoterAccess | A user can get access to his data but not to other user's data | El acceso a los datos 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 y ReportWriter | 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 y ReportWriter | 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 | Los desarrolladores pueden implementar el sistema | El sistema puede implementarse en 2 semanas | **AT013** |
| 14 | Administrador del Sistema | Despliega el sistema | Despliegue | Censuses, VoterInfo | El sistema es desplegado en un entorno de producción | El sistema puede desplegarse en menos de una hora | **AT014** |
|  |  |  |  |  |  |  |  |

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 |

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

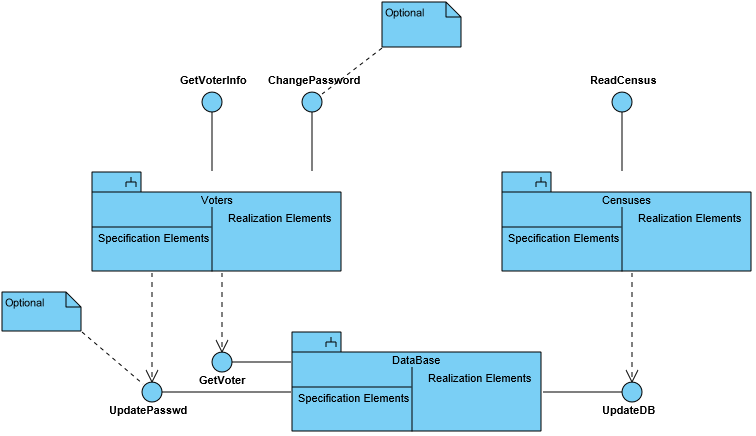


Figura 3. Context view

### 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. |

#### 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*.

#### 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. |

#### 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

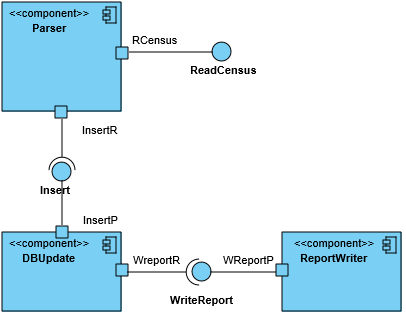


Figura 4. Vista Census Reader

### 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

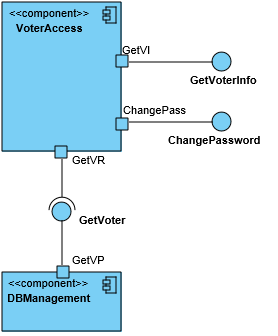


Figura 5. Vista de VoterInfo

### 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. |

# 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.