Sistema de Gestión de Servicio Comunitario (SIGESC) de la Universidad Simón Bolívar

Documento de la Arquitectura del Software

Versión 3.0

**Historial de Revisión**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| <10/03/16> | <1.0> | <Compendio de documentos> | <RupDev> |
| <28/04/16> | <1.1> | <Continuación de desarrollo> | <SLEEK Software> |
| <12/05/16> | <1.2> | <Actualizaciones y correcciones sobre 1era iteración de construcción> | <SLEEK Software> |
| <02/06/2016> | <2.0> | <Actualizaciones y correcciones sobre 3era iteración de construcción> | <SLEEK Software> |
| <02/06/2016> | <3.0> | <Actualizaciones y correcciones sobre 4ta iteración de construcción> | <SLEEK Software> |

Tabla de Contenidos.

1. Introducción 4

1.1 Propósito 4

1.2 Alcance 4

1.3 Definiciones, Siglas, y Abreviaciones 4

1.4 Referencias 4

1.5 Vista Global 5

2. Representación Arquitectónica 5

3. Metas y Restricciones Arquitectónicas 5

4. Vista de Casos de Uso 7

5. Vista Lógica 10

5.1 Visión general 10

5.2 Paquetes de Diseño Significativos Arquitectónicamente 11

5.3 Realizaciones de los casos de uso 12

5.4 Diagrama de Estados 15

5.5 Diagramas WAE 16

6. Vista de Implantación 27

7. Vista de Implementación 28

8. Vista de Datos 29

8.1 Diagrama ERE 29

8.2 Diccionario de Datos 30

9. Tamaño y Desempeño 34

10. Calidad 34

# **Introducción**

## **Propósito**

Mediante este documento se quiere presentar una visión general de la arquitectura del SIGESC, para el cual se usarán distintas vistas arquitectónicas que permitan representar distintos aspectos del sistema. El mismo reúne y comunica las decisiones arquitectónicas importantes que se han hecho sobre el proyecto, brindando al lector una visión global y comprensible del diseño general de este.

El modelado de las vistas permitirá crear un esquema del sistema mucho más completo y darle así una base sólida para su desarrollo. En la versión inicial, incluye la vista de casos de Uso con el diagrama de casos de uso, la vista lógica con el modelo conceptual y el diagrama de clases y la vista de datos con el diagrama ER y el diccionario de datos.

## **Alcance**

Este documento se centra en las vistas lógicas, casos de uso y de datos del sistema; mencionando los aspectos importantes de la vista de implantación e implementación sin tocar la vista de procesos al no corresponder a este proyecto.

Entre los documentos directamente influenciados por éste se encuentran la lista de riesgos (al comprender como interactúan los casos de uso pueden surgir riesgos sobre estas conexiones) y los planes de iteración (ya que el DAS debe guiar las decisiones de implementación).

**1.3 Definiciones, Siglas, y Abreviaciones**

* **RUP**: Rational Unified Process. Se trata de una metodología para describir el proceso de desarrollo de software.
* **UML**: Unified Modeling Language.
* **USB**: Universidad Simón Bolívar.
* **CFCG**: Coordinación de Formación Complementaria General.
* **SIGESC**: Sistema de Gestión del Programa de Servicio Comunitario.
* **DEx**: Decanato de Extensión.
* **SC**: Servicio Comunitario.
* **DII**: Dirección de Ingeniería de la Información.
* **SCA**: Servicio Centralizado de Autenticación.
* **ERS**: Especificación de Requerimientos de Software.

## **1.4 Referencias**

* Plantilla del documento de la arquitectura del software.
* RUP (Rational Unified Process).
* Diagrama de casos de uso de SIGESC.
* Documento de Visión de SIGESC

**1.5 Vista Global**

Los capítulos restantes constan de la representación arquitectónica, donde se describe la arquitectura utilizada en SIGESC; las metas y restricciones, donde se describen los objetivos a alcanzar en el desarrollo del proyecto y todos los aspectos que lo delimitan, como las herramientas a usar, los recursos disponibles, código legado y tiempos de entrega.

Luego vienen las vistas, donde la de casos de uso expone el diagrama de los mismos en una versión general y varias detalladas por actor y paquetes. La lógica muestra las interacciones entre los distintos actores con el modelo de dominio y entre los casos de uso críticos con los diagramas de secuencia.

La vista implantación muestra las configuraciones de hardware en las cuales se monta y desarrolla el sistema. La vista de implementación presenta la configuración de las capas del esquema o diseño utilizado. Finalmente, la vista de datos se enfoca en el almacenamiento de los datos a través del modelo Entidad Relación. Los últimos segmentos del documento se enfocan en el tamaño y aspectos de calidad del sistema.

# **2. Representación Arquitectónica**

Para el desarrollo de SIGESC se adopta el modelo 4+1 de Kruchten, el cual permite describir la arquitectura en múltiples vistas. Dichas vistas se usan para describir el sistema desde los puntos de vista de distintos “stakeholders”, tales como los usuarios finales, desarrolladores y administradores del proyecto. Los 5 modelos son:

Vista lógica: se enfoca en las funcionalidades que el sistema proporciona a usuarios finales. Principalmente se usan los diagramas UML para el modelado.

Vista de procesos: detalla los aspectos dinámicos, explica los procesos y cómo se comunican. Esta vista hace énfasis en el comportamiento de la corrida del sistema.

Vista de desarrollo: ilustra el sistema desde el punto de vista de los programadores y se centra en el manejo del software.

Vista física: muestra el sistema desde el punto de vista del ingeniero. Se concentra en el hardware utilizado y sus conexiones.

Escenarios: se muestra la arquitectura del software a través de los casos de uso; éstos muestran la interacción entre los distintos actores y las funciones del sistema.

# **3. Metas y Restricciones Arquitectónicas**

La meta de este proyecto es elaborar un sistema para la gestión del servicio comunitario que facilite los trámites utilizados en la actualidad; debe ser eficaz en el manejo de los archivos referentes a cada proyecto e involucrados, rápido en los tiempos de respuesta para las distintas solicitudes y confiable en el almacenamiento de los datos.

Las restricciones que dirigen el desarrollo de SIGESC son:

* Usar el lenguaje de programación Python
* Implementar los módulos con Web2py
* Usar PostgreSQL como manejador de base de datos.
* Regirse por el esquema MVC para la implementación.
* Trabajar bajo el sistema RUP a lo largo del trimestre.
* Entregar 77% de las funcionalidades para el 24 de junio de 2016
* Guiarse por el código legado por Quantum, Rupdev, Syscorp y RoraimaTech.

# 

# **4.** **Vista de Casos de Uso**

# 

# 

# 

# 

# 

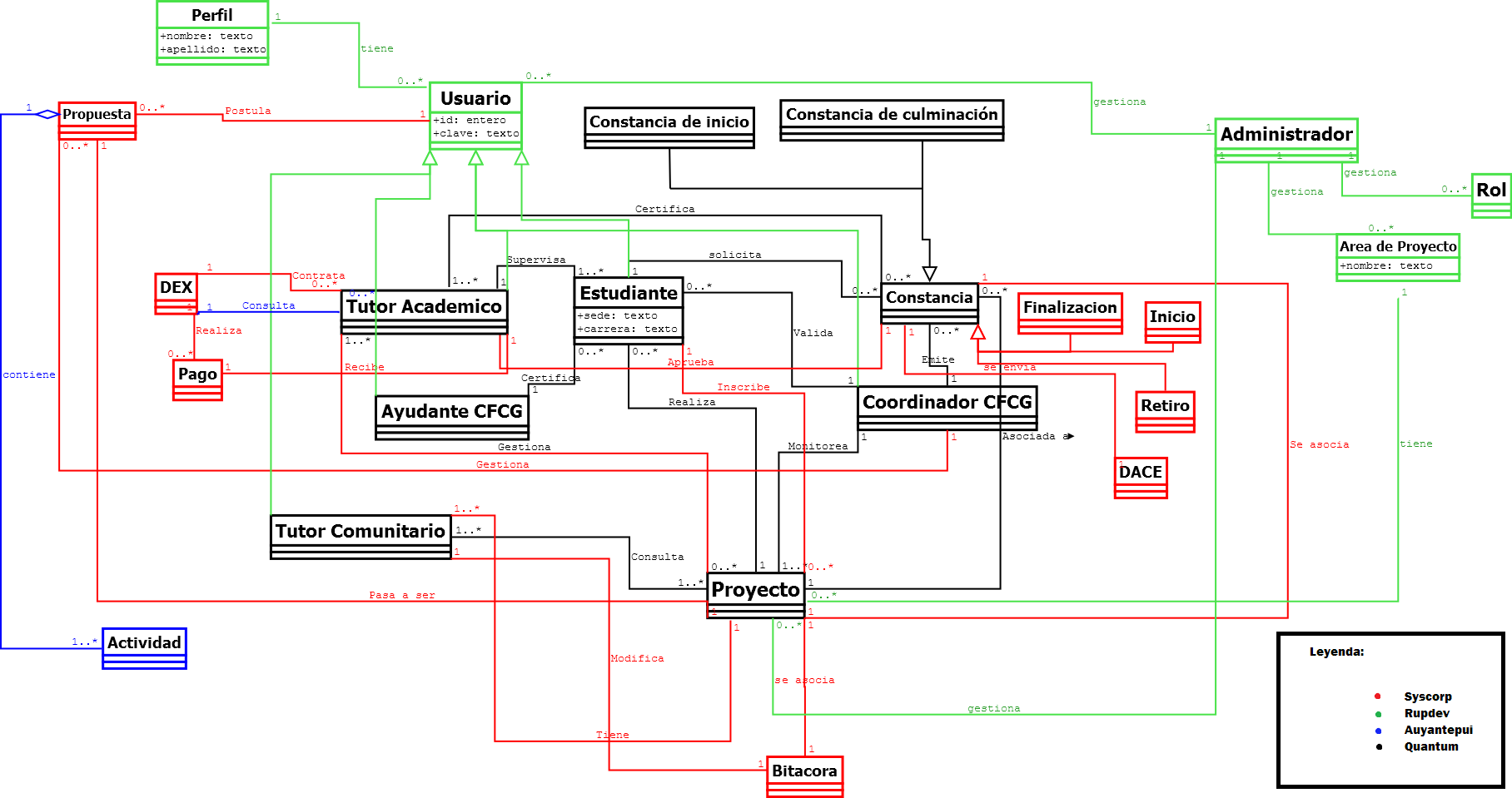
# 

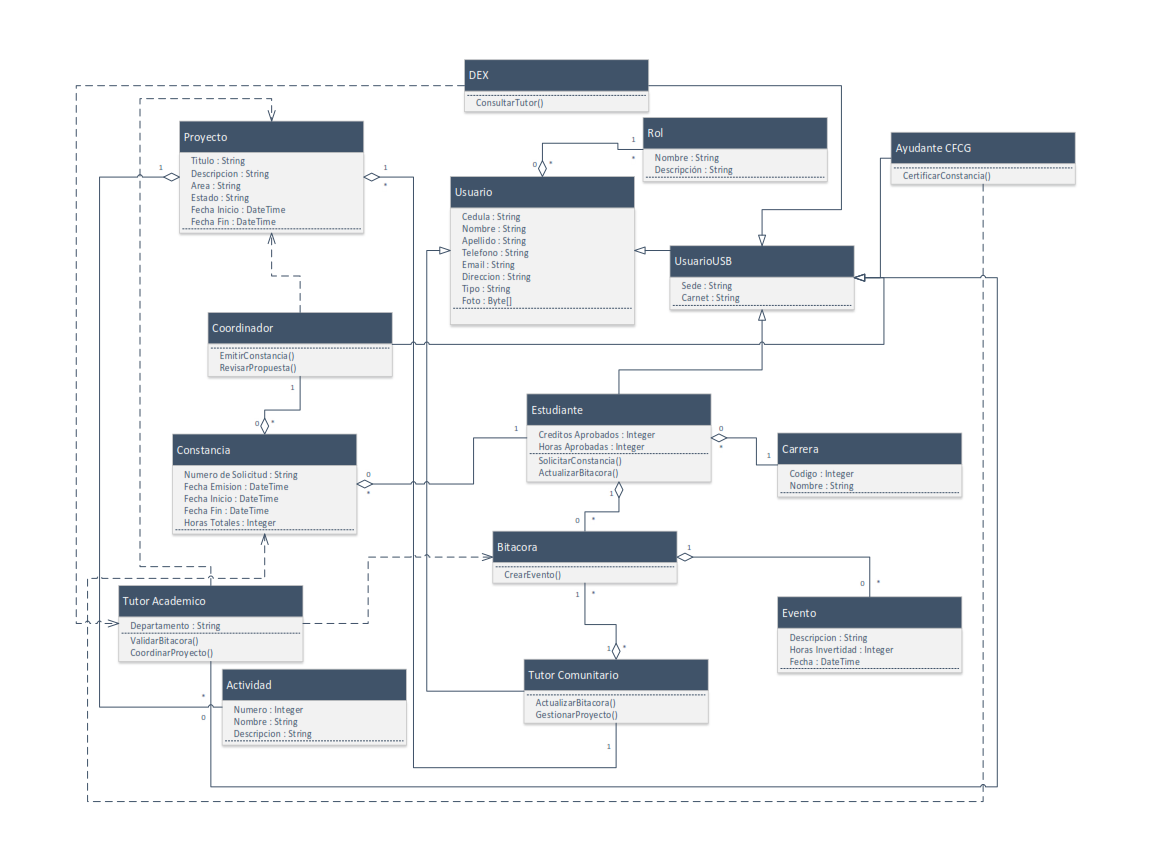
# 

# 

# **5.** **Vista Lógica**

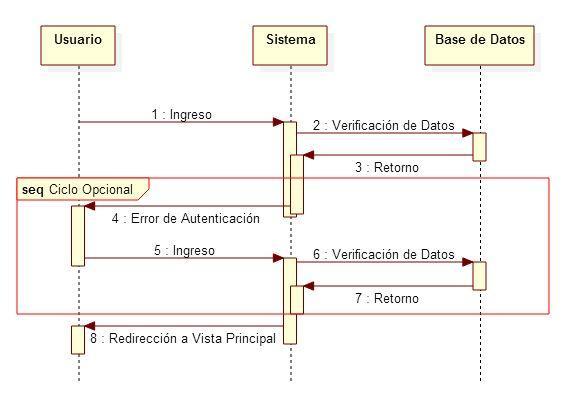
**5.1 Visión General**



**5.2 Paquetes de Diseño Significativos Arquitectónicamente.**

**5.3 Realizaciones de los Casos de Uso**

* Iniciar Sesión sin USB-ID:



1. El usuario ingresa su usuario y contraseña.
2. El sistema verifica los datos con la Base de Datos.
3. La Base de Datos retorna la verificación de los datos.

**Ciclo Opcional (4-7): En caso de un error de autenticación**

1. El sistema envía un mensaje al Usuario indicándole un error de autenticación.
2. El usuario vuelve a ingresar su usuario y contraseña.
3. El sistema verifica los datos con la Base de Datos.
4. La Base de Datos retorna la verificación de los datos.

**Fin de Ciclo Opcional.**

1. El sistema re direcciona al usuario a la vista principal según su rol.

### Iniciar Sesión con USB-ID:

1. El usuario ingresa su usuario y contraseña.
2. El sistema verifica los datos con CAS
3. CAS retorna la verificación de los datos.

**Ciclo Opcional (4-7): En caso de un error de autenticación**

1. El sistema envía un mensaje al Usuario indicándole un error de autenticación.
2. El usuario vuelve a ingresar su usuario y contraseña.
3. El sistema verifica los datos con CAS
4. CAS retorna la verificación de los datos.

**Fin de Ciclo Opcional.**

1. El sistema re direcciona al usuario a la vista principal según su rol.

### Registrar Usuario sin USB-ID:

1. El visitante solicita registro.
2. El sistema muestra formulario.
3. El visitante rellena formulario.
4. El sistema verifica si el formulario tiene datos válidos con la base de datos (Usuario ya existe, tipo de datos inválidos, datos vacíos)
5. La base de datos retorna la verificación de los datos.

**Ciclo Opcional (6-7): En caso de un error en el formulario**

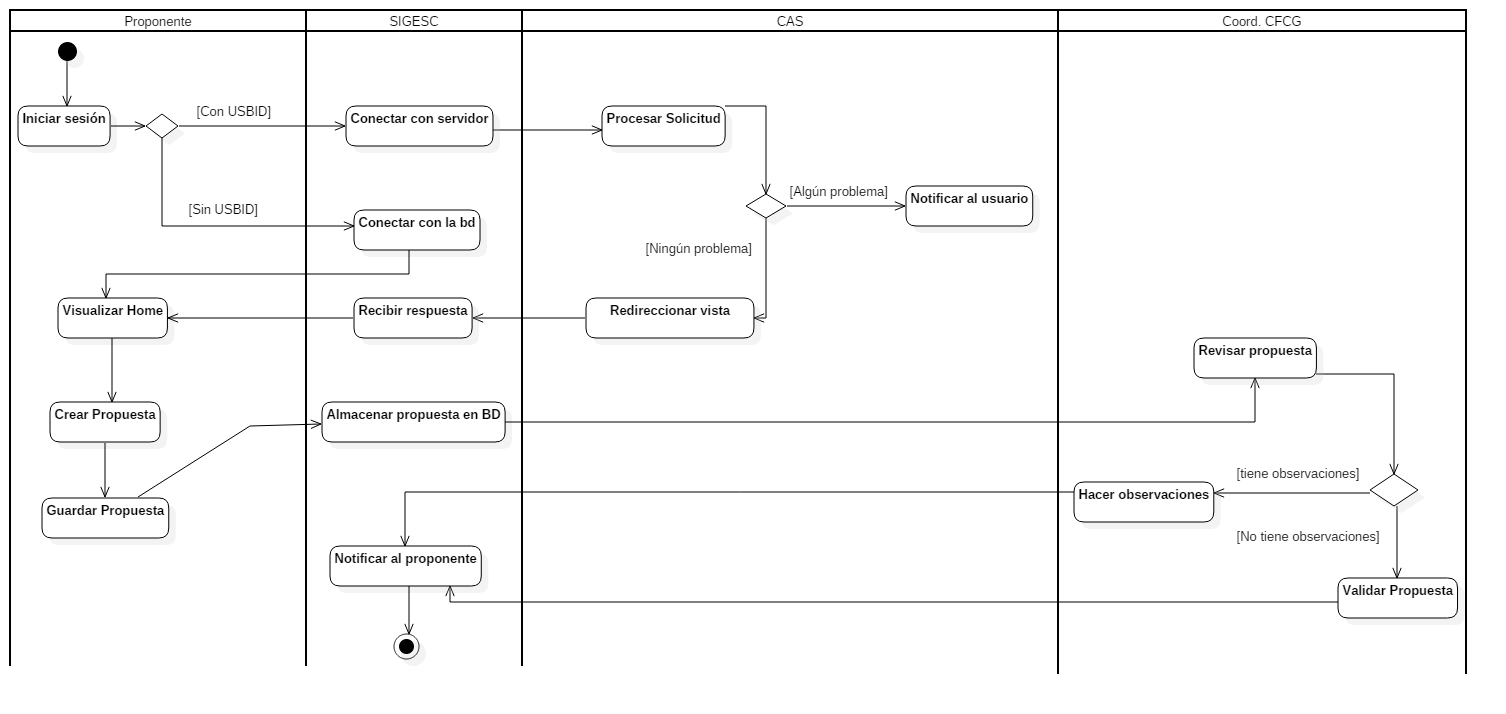
1. El sistema envía un mensaje al Usuario indicándole el error del formulario.
2. El usuario vuelve a rellenar el formulario.
3. El sistema verifica si el formulario tiene datos válidos con la base de datos
4. La base de datos retorna la verificación de los datos.

**Fin de Ciclo Opcional.**

1. El sistema registra el usuario en la base de datos.
2. Se le muestra un mensaje de registro exitoso.

**5.4 Diagrama de Estados.**

Este diagrama se utiliza para describir el comportamiento del sistema de gestión de servicio comunitario, por lo que se describe todos los estados posibles en los que puede entrar al proponer una propuesta y la manera en que cambia el estado de la misma, como resultado de los eventos que llegan a ella.

****

# 

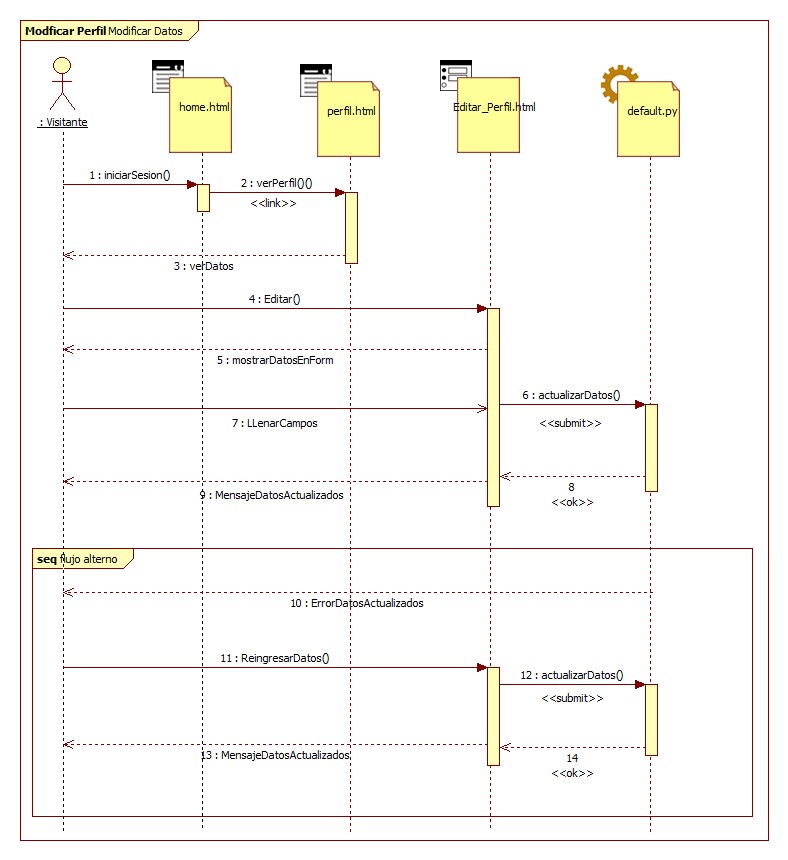
# 

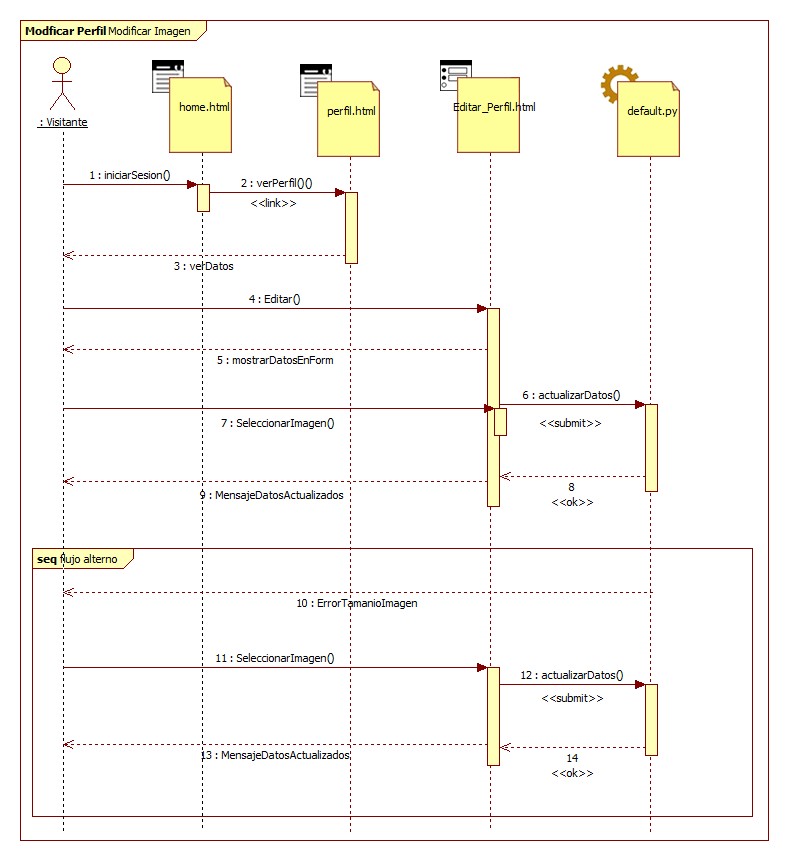
**5.5 Diagramas WAE**

* **Registro (Usuario Visitante)**

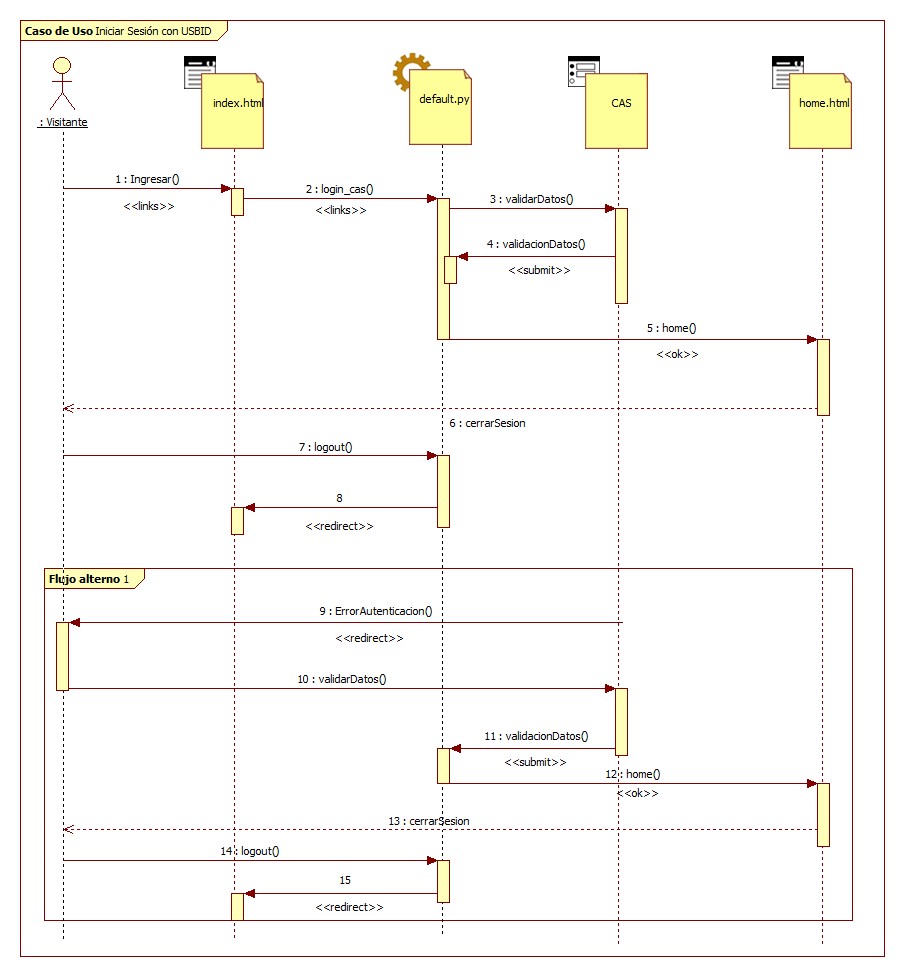
****

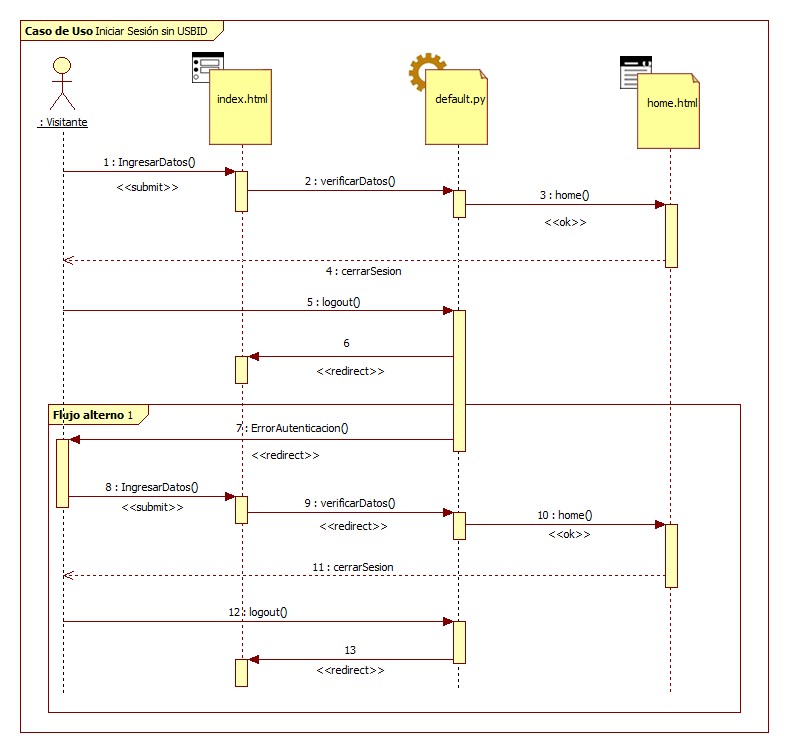
* **Modificar Perfil- Datos e Imagen (Usuario Proponente)**

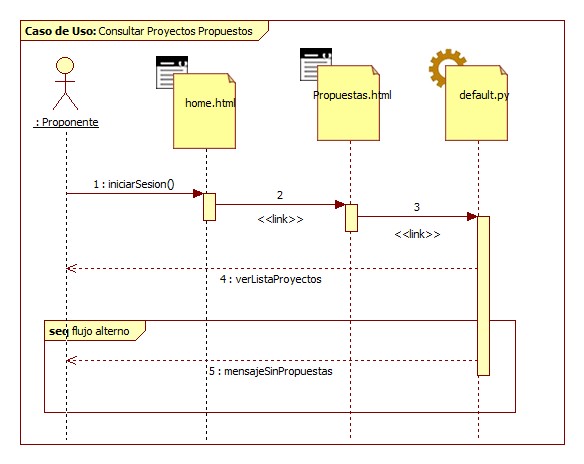
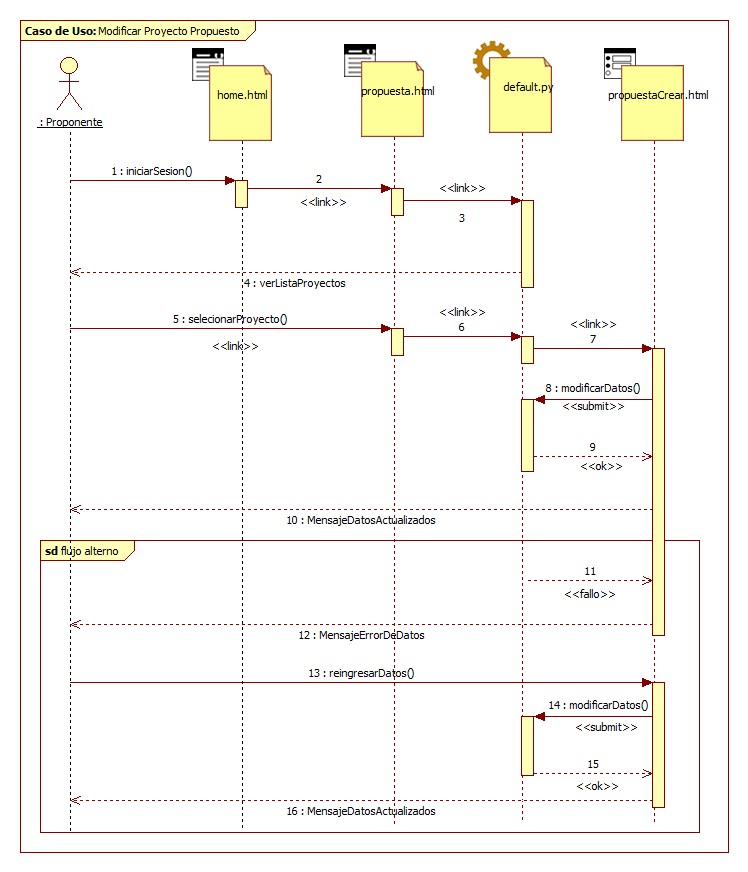
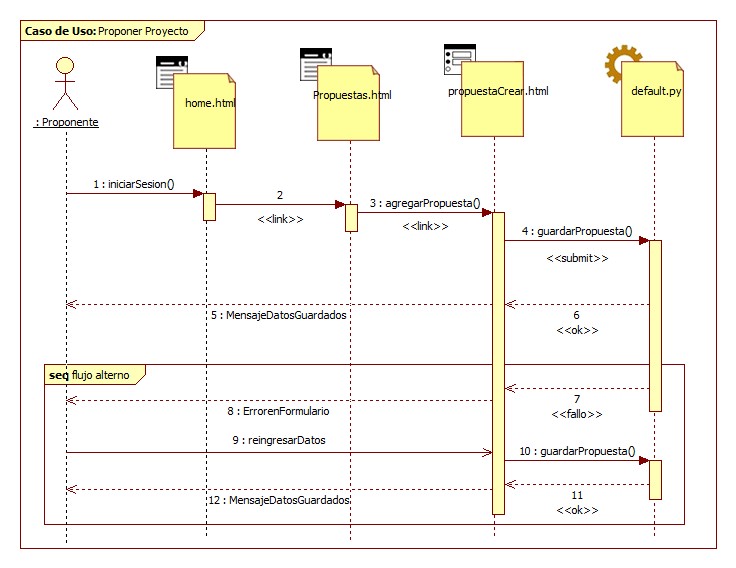


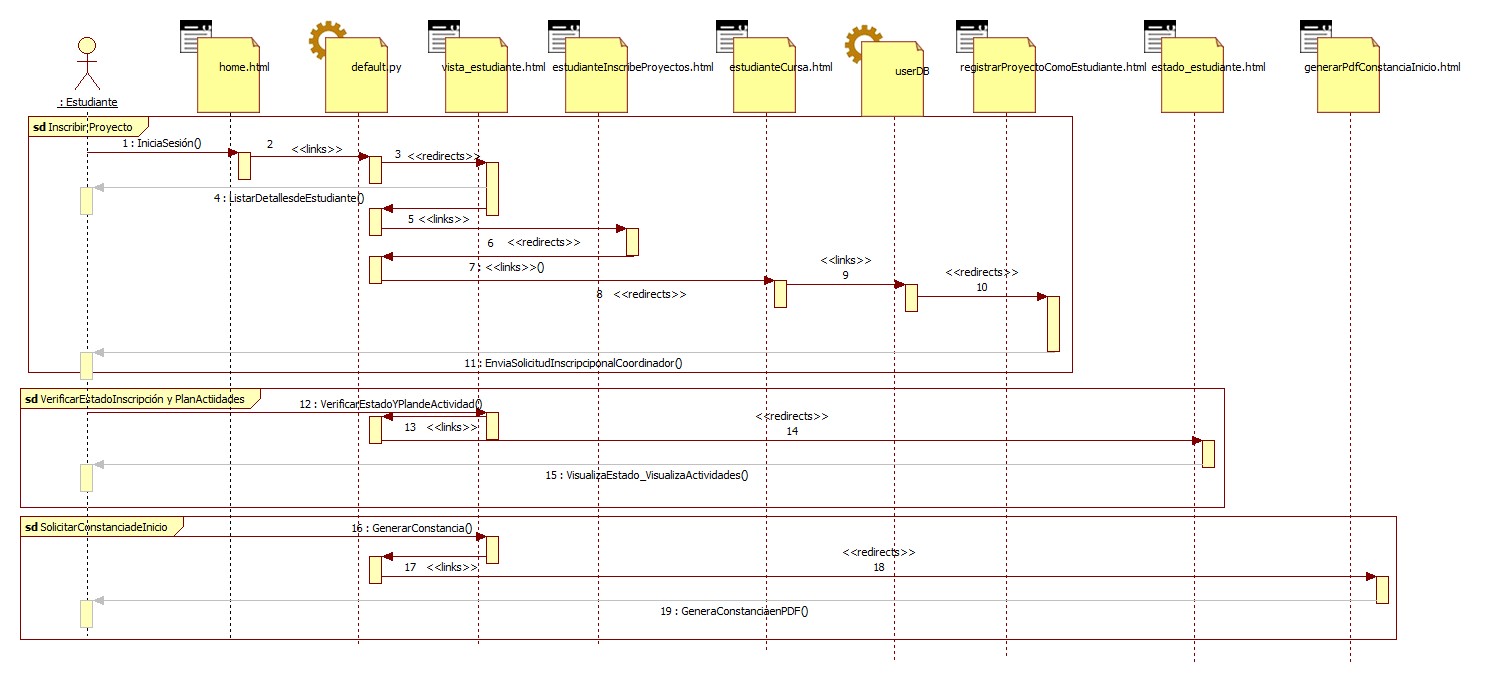


* **Iniciar Sesión con y sin USBID (Usuario Proponente)**

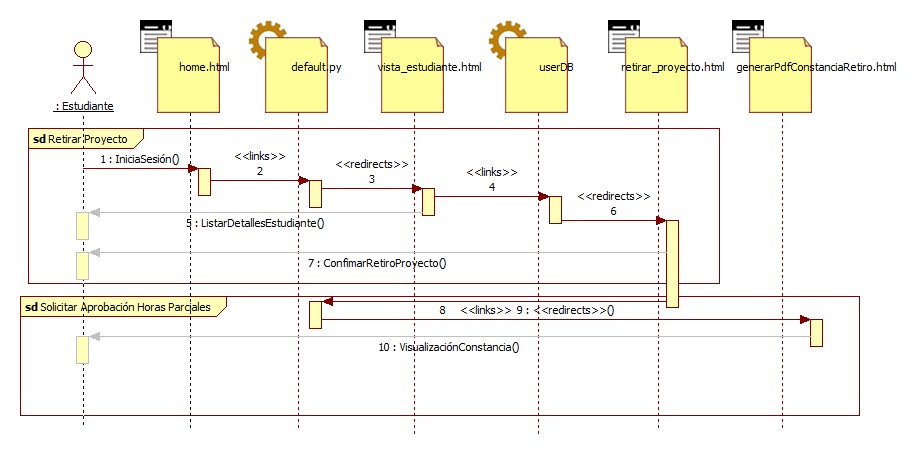
****

****

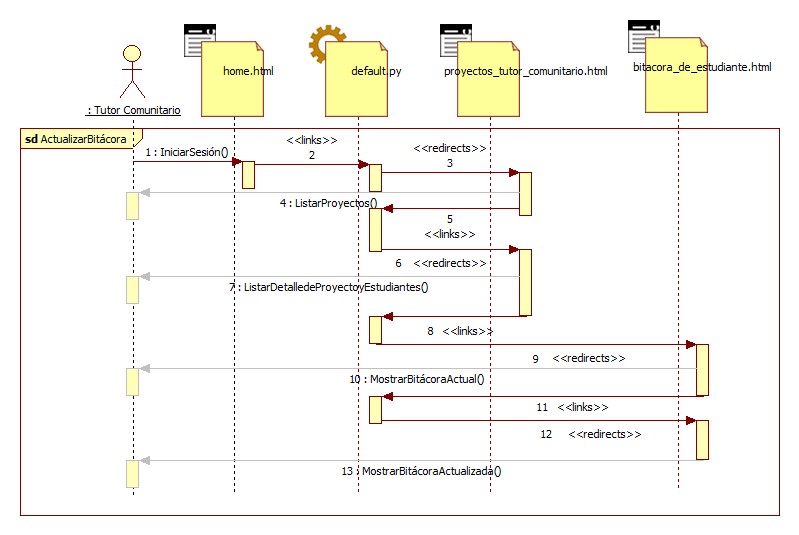
* **Gestionar Proyecto-Proponer-Modificar-Consultar (Usuario Proponente)**
* **Inscribir Proyecto -(Estudiante)**

****

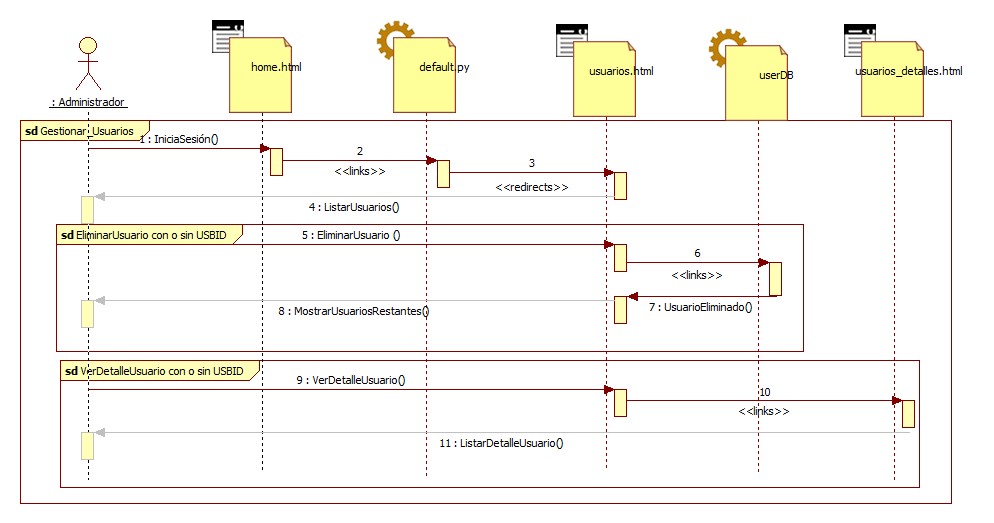
* **Retirar Proyecto (Estudiante)**

****

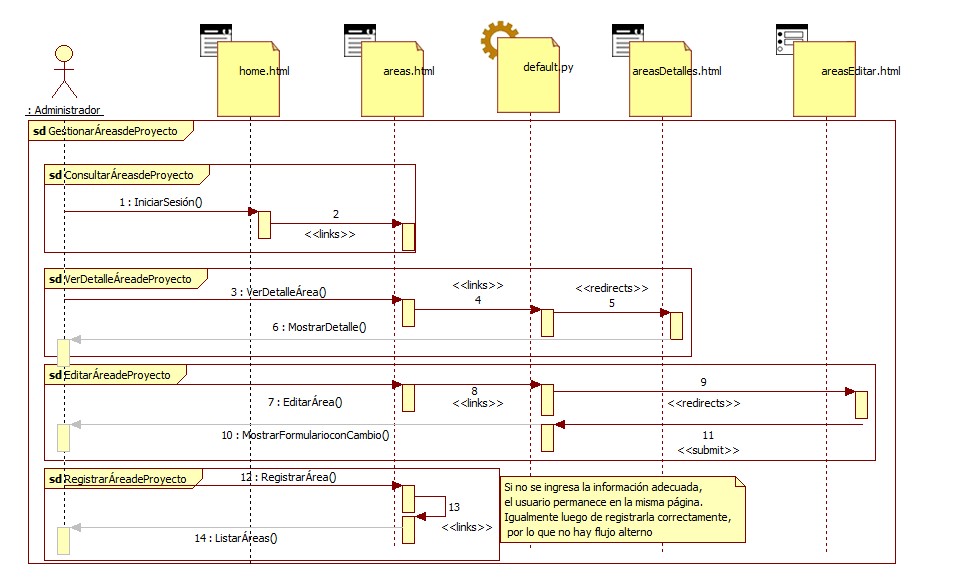
* **Actualizar Bitácora (Tutor Comunitario)**

****

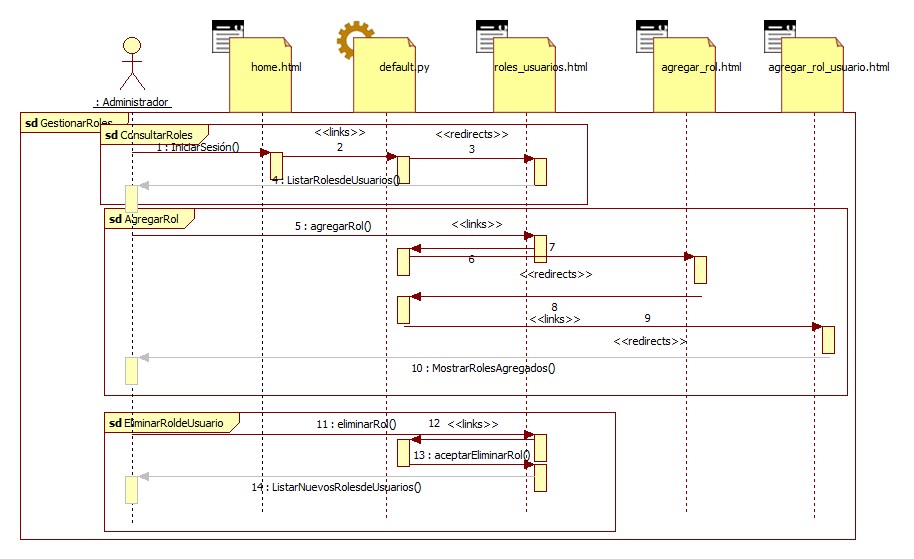
* **Gestionar Usuario (Adm de Mantenimiento)**

****

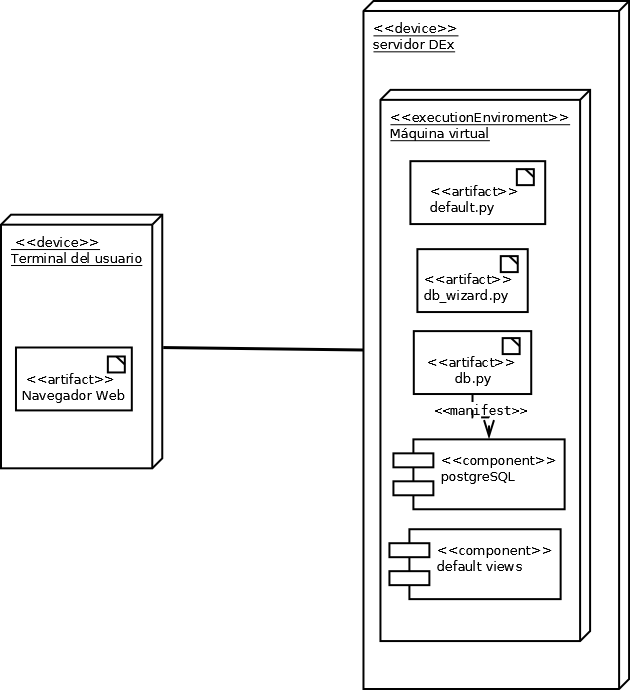
* **Gestionar Áreas de Proyecto (Adm de Mantenimiento)**

****

* **Gestionar Roles (Adm de Mantenimiento)**

****

# **6.** **Vista de Implantación**



# 

# **7. Vista de Implementación**

Se implementa el patrón modelo-vista-controlador (MVC). En este se establecen tres capas de división: la capa modelo, la capa vista y la capa controlador.

Se presenta a continuación el diagrama resumido de componentes del sistema, donde el administrador de direcciones asocia la URL solicitada a una llamada a función en el controlador. La salida de la función puede ser una cadena o un diccionario conteniendo pares nombre-valor (una *hash table*). Los datos contenidos en el diccionario se convierten en la página HTML. Si el visitante solicita la misma página en XML, web2py busca una vista que pueda convertir el diccionario a XML. El desarrollador puede crear vistas que conviertan páginas en cualquier protocolo soportado (HTML, XML, JSON, RSS, CSV y RTF) o en protocolos personalizados adicionales.

Finalmente todas las llamadas se envuelven en una transacción, y toda excepción no manejada hace que la transacción recupere el estado inicial. Si la solicitud tiene éxito, se aplican los cambios a la base de datos

# 

# **8. Vista de Datos**

**8.1 Diagrama ERE**

# EREunificado.png

**8.2 Diccionario de datos:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entidad** | **Semántica** | **Atributos** | **Semántica del Atributo** | **Dominio** |
| AYUDANTE\_CFCG | Ayudante de la Coordinación de Formación Complementaria General | ci | Cédula de identidad del ayudante | Secuencia de enteros |
| nombre | Nombre del ayudante | Secuencia de caracteres |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entidad** | **Semántica** | **Atributos** | **Semántica del Atributo** | **Dominio** |
| CONSTANCIA | Constancia emitida por la coordinación | carnet | Carnet del estudiante que solicita la constancia | Secuencia de enteros |
| fecha | Fecha en la cual es emitida la constancia | Datetime |
| nom\_  solicitante | Nombre del estudiante que solicita la constancia | Secuencia de caracteres |
| num\_  solicitud | Número de la solicitud de la constancia | Secuencia de enteros |
| CONSTANCIA\_INICIO | Constancia de inicio que valida la inscripción del estudiante | fecha\_ini | Fecha de la constancia de inicio del estudiante | Datetime |
| CONSTANCIA\_FIN | Constancia de finalización del servicio comunitario | cant\_hrs | Cantidad de horas realizadas del servicio comunitario por parte del estudiante | Secuencia de enteros  (mayor 40 hrs) |
| fecha\_fin | Fecha de finalización del servicio comunitario | Datetime |
| nom\_tutor | Nombre del tutor del servicio comunitario | Secuencia de caracteres |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entidad** | **Semántica** | **Atributos** | **Semántica del Atributo** | **Dominio** |
| USUARIO | Usuario del sistema SIGESC | id | Identificador | Id |
| nombre | Nombre de usuario | Secuencia de caracteres |
| correo\_elect | Email del usuario | Secuencia de caracteres |
| clave | Contraseña del usuario para ingresar al sistema | Secuencia de caracteres |
| tipo | Tipo de usuario (universitario o externo) | Secuencia de caracteres |
| dirección | Dirección del usuario | Secuencia de caracteres |
| cedula | Cédula del usuario | Secuencia de caracteres |
| teléfono | Teléfono del usuario | Secuencia de caracteres |
| foto | Foto del usuario | Archivo |
| USUARIO UNIVERSITARIO | Usuario universitario que posee USBID | carnet | Carnet del usuario | Secuencia de caracteres |
| sede | Sede a la que pertenece el usuario | Secuencia de caracteres |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entidad** | **Semántica** | **Atributos** | **Semántica del Atributo** | **Dominio** |
| ROL | Roles de usuarios del sistema (administrador, coordinador, asistente, etc) | nombre | Nombre rol | Secuencia de caracteres |
| descripción | Descripción del rol | Secuencia de caracteres |
| id | Identificador del tipo de rol | Id |
| ESTUDIANTE | Estudiante que inscribe un proyecto de servicio comunitario | créditos\_  aprobados | Número de créditos aprobado por el estudiante | Secuencia de enteros |
| carrera | Carrera que cursa el estudiante | Secuencia de caracteres |
| hrs\_aprobadas | Cantidad de horas aprobadas de la realización del servicio comunitario | Secuencia de entero  (mayor a 40 hrs) |
| TUTOR COMUNITARIO | Responsable del servicio comunitario en la comunidad beneficiada |  |  |  |
| BITÁCORA | Registro de eventos del servicio comunitario | fecha | fecha del evento registrado en la bitácora del proyecto | Fecha y hora |
| evento | Nombre del evento registrado en la bitácora del proyecto | Secuencia de caracteres |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entidad** | **Semántica** | **Atributos** | **Semántica del Atributo** | **Dominio** |
| (BITÁCORA)  EVENTO | Evento registrado en la bitácora del servicio comunitario | descripción | Descripción del evento | Secuencia de caracteres |
| horas\_  invertidas | Horas invertidas del estudiante en el evento | Secuencia de enteros |
| TUTOR ACADÉMICO | Tutor académico de un proyecto de servicio comunitario | departamento | Departamento al que pertenece el tutor académico | Secuencia de caracteres |
| COORDINADOR | Coordinador de la CFCG |  |  |  |
| PROYECTO | Proyecto de Servicio Comunitario | titulo | Título de la propuesta | Secuencia de caracteres |
| descripcion | Descripción de la propuesta | Secuencia de caracteres |
| fecha\_inicio | Fecha tentativa para iniciar el proyeto | Fecha |
| fecha\_  terminación | Fecha tentativa para el proyecto | Fecha |
| area | Área en la que está orientada la propuesta | Secuencia de caracteres |
| estado | Estado de la propuesta | Secuencia de caracteres |
| PROPUESTA | Propuesta de Proyecto de Servicio comunitario |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entidad** | **Semántica** | **Atributos** | **Semántica del Atributo** | **Dominio** |
| APROBADO | Propuesta de Proyecto de Servicio comunitario aprobada por el coordinador de la CFCG |  |  |  |
| ACTIVIDAD | Actividad a desarrollarse en la propuesta de proyecto | numero | Número de la actividad | Entero > 0 |
| nombre | Nombre de la actividad a desarrollar | Secuencia de caracteres |
| resumen\_act | Descripción de la actividad | Secuencia de caracteres |
| alumnos | Número de alumnos que se requieren para llevar a cabo la actividad | entero > 0 |
| requerimiento | Especificación o necesidad especial de los alumnos para esta actividad | Secuencia de caracteres |

# **9. Tamaño y Desempeño**

* Se necesita acceso a internet y la presencia de un navegador web para que un usuario pueda usar el sistema.
* Cualquiera de los actores debe poder utilizar el sistema de manera simultánea.
* Debe tener la capacidad de indicar a los usuarios cuando se encuentre fuera de servicio.
* Se debe generar un comprobante del proceso realizado por un usuario en formato .pdf

# **10. Calidad**

* El uso del patrón MVC mantiene el orden de la codificación por su funcionalidad. Además, al estandarizar el desarrollo se facilita su modificación en el futuro por distintos programadores.
* El uso de una interfaz sencilla facilita la navegación del usuario por las vistas del sistema.
* Al desarrollarse con web2py se facilita la implementación del patrón MVC y al usar Python vuelve al sistema accesible y duradero en el tiempo para los desarrolladores y encargados de su mantenimiento.
* Con el uso de RUP se asegura una continua revisión, evaluación y mitigación de los riesgos que surgen a lo largo del desarrollo.
* Establecer postgreSQL como manejador de base de datos da confiabilidad sobre el almacenamiento de los datos, también asegura que las operaciones hechas en alto tráfico de datos sean realizadas correctamente.