



Universidad Austral de Chile

Facultad de Ciencias de la Ingeniería

Escuela de Ingeniería Civil en Informática

SISTEMA DE APOYO AL MONITOREO CURRICULAR DE ESTUDIOS DE PREGRADO UACH

Proyecto para optar al título de
Ingeniero Civil en Informática

PROFESOR PATROCINANTE:
MAURICIO RUIZ-TAGLE MOLINA
INGENIERO CIVIL EN INFORMÁTICA
DOCTORADO EN SOFTWARE Y SISTEMAS

PROFESOR INFORMANTE:
MARIANNA VILLARROEL MANFREDI
INGENIERO CIVIL EN INFORMÁTICA
MAGISTER EN EDUCACIÓN

PROFESOR INFORMANTE:
JORGE MORALES VILUGRON
INGENIERO EN ELECTRICIDAD

BALDOMERO ÁGUILA NAPOLI

VALDIVIA - CHILE

2015

AGRADECIMIENTOS

En primer lugar quiero agradecer a la mujer más importante de mi vida, la que se desveló más de una noche para cuidar de mi salud, la que muchas veces pasó hambre para que yo me alimentara bien, la mujer que me enseñó a levantarme todas las veces que sean necesarias para lograr mis objetivos, estoy hablando de mi madre, María Napoli, sin ella yo no estaría escribiendo este documento, gran parte de mis logros se los debo a ella. Agradezco a mi hermano, que a pesar de todo lo ocurrido él me enseñó que en la vida hay que ser valiente, él me enseñó a que pase lo que pase, nunca agachar la cabeza. También quiero agradecer a mis hermanas, Mavian y Alejandra por su incondicional apoyo.

Quiero agradecer de forma especial a mis dos grandes amigos que siempre me apoyaron y creyeron en mí, Tere y Colin, sin ustedes no me hubiera levantado tan fácilmente, ustedes han bailado conmigo bajo la lluvia, y no lo digo tan literalmente.

Agradezco a mi tía nana, que al momento de pedirle ayuda, no dudó en ofrecermé más de lo que yo le pedí.

Agradezco a mi segundo hogar, el Trensito, con ustedes conviví más de cinco años, y fue genial sentir el compañerismo que existe ahí.

César Soto, a pesar de que nos conocimos en un ambiente laboral, te convertiste en un gran amigo y siempre estuviste dispuesto a ayudarme, te lo agradezco de todo corazón, no muchas personas son como tú.

Con respecto a los profesores de la universidad, quiero agradecer de forma especial al profesor Mauricio, quien se convirtió en mi profesor guía durante la última etapa de mi formación profesional y a la profesora María Eliana, quien siempre me brindó su ayuda, a pesar de que estuviera ocupada.

Quiero agradecer de forma especial a una persona que no nombraré, sin embargo dicha persona apareció al final de esta etapa, ella me retó, me saco muchas veces de mi estrés, me recordó lo que era amar, te agradezco infinitamente todo lo que hiciste por mí M.V.C.LI!!! Por último quiero agradecer a todas las personas que no he podido nombrar, pero que de alguna u otra manera me han ayudado a finalizar este proceso, GRACIAS A TODOS!!!.

ÍNDICE

ÍNDICE	I
ÍNDICE DE TABLAS	III
ÍNDICE DE FIGURAS	IV
RESUMEN	V
ABSTRACT	VII
1 INTRODUCCIÓN	1
1.1 Definición del Proyecto	1
1.1.1 Objetivo general	1
1.1.2 Objetivos específicos	2
1.2 Nivel actual	2
1.2.1 Departamento de Aseguramiento de la Calidad e Innovación Curricular (DACIC)	2
1.2.1.1 Apoyo al Desarrollo y la Innovación Curricular en Pregrado	3
1.2.2 Departamento de Admisión y Matrícula	4
1.2.2.1 Servicios estudiantiles	4
1.2.3 Departamento de Registro Académico Estudiantil	4
1.2.3.1 Procesos	4
1.2.3.2 Creación de nuevas Carreras	5
1.2.3.3 Modificaciones Curriculares Mayores y/o Menores	7
1.3 Metodología	9
2 MARCO TEÓRICO	10
2.1 Arquitectura	10
2.1.1 Pseudo MVC	10
2.1.1.1 Biblioteca de servicios WCF	12
2.2 Tecnologías para el desarrollo de la plataforma	13
2.2.1 Front-end	13
2.2.1.1 JQuery	14
2.2.1.2 Alertify	14
2.2.1.3 Bootstrap	15
2.2.1.4 Template Metis	15
2.2.1.5 Parsley	15
2.2.2 Back-end	16
2.2.2.1 Microsoft SQL Server 2008	16
2.2.2.2 ASP.NET	16
2.2.3 Herramientas Anexas	17
2.2.3.1 GitHub	17
2.2.3.2 Firebug	18
2.3 Tipos de documentos	19
3 DESARROLLO DEL SISTEMA	20
3.1 Planificación	20
3.1.1 Metodología	20
3.1.2 Plan de trabajo	21
3.2 Análisis	22
3.2.1 Requerimientos	22
3.2.1.1 Requisitos funcionales	22
3.2.1.2 Requisitos no funcionales	24
3.2.2 Modelo conceptual	24
3.2.3 Casos de uso	25
3.2.3.1 Descripción de los actores del sistema	28
3.3 Diseño	29
3.3.1 Diagramas de componentes	29
3.3.2 Modelo de datos	30
3.3.2.1 Diccionario de datos	31
3.3.3 Módulo historial curricular	31
3.3.3.1 Diseño	32
3.3.4 Módulo Gestor de Documentos	36
3.3.4.1 Diseño	36
4 IMPLEMENTACIÓN	46
4.1 Configuración del ambiente de desarrollo	46

4.2	Consideraciones Técnicas	46
4.3	Dificultades de la implementación	47
4.4	Presentación de prototipos	48
5	VALIDACIÓN DE LA PLATAFORMA	49
5.1	REQF-01: Autenticación de usuario	49
5.2	REQF-02: Gestión de perfil	50
5.3	REQF-03: Desplegar historial curricular	52
5.4	REQF-04: Registro de usuarios	52
5.5	REQF-05:Gestión de documentos	53
5.6	REQF-06:Visor de PDF	54
5.7	REQF-07:Notificaciones	54
5.8	REQF-08:Almacenar bitácora de los usuarios	54
5.9	REQF-09:Visualización de bitácora	55
6	CONCLUSIONES	56
6.1	Trabajo futuro	57
7	REFERENCIAS	59
8	ANEXOS	61
	Anexo A: Formato inicial de los archivos JSON.	61
	Anexo B: CascadingDropDown	61
	Anexo C: Diccionario de datos.	62

ÍNDICE DE TABLAS

TABLA	PÁGINA
1 Requisitos funcionales	22
2 Requisitos no funcionales	24
3 Caso de uso Ver historial curricular	32
4 Caso de uso Subir Documento	37
5 Caso de uso Ver documentos	42
6 Resultados de pruebas del acceso a la gestión de perfiles	51
7 Propuesta de trabajo	57

ÍNDICE DE FIGURAS

FIGURA	PÁGINA
1 Organigrama Vicerrectoría y DACIC	3
2 Proceso de presentación de nuevos proyectos	6
3 Procesos de cambios Curriculares Mayores y/o menores	8
4 Arquitectura pseudo MVC Universidad Austral de Chile	11
5 Diagrama de componentes de un servicio WCF	13
6 Carta Gantt que exhibe las etapas de desarrollo del sistema	21
7 Modelo conceptual del proyecto	25
8 Diagrama de caso de uso para el Administrador	26
9 Diagrama de caso de uso para el Editor	27
10 Diagrama de caso de uso para el Suscriptor	28
11 Diagrama de componentes Software del sistema	29
12 Diagrama de componentes Hardware del sistema	30
13 Modelo de datos Entidad-Relación	31
14 Diagrama de secuencia para el caso de uso <i>Ver Historial Curricular</i>	35
15 Diagrama de secuencia para el caso de uso <i>Subir Documentos</i>	41
16 Diagrama de secuencia para el caso de uso <i>Ver Documentos</i>	44
17 REQF-01: Autenticación de usuario	49
18 REQF-01: Autenticación de usuario, validación de campos nulos	50
19 REQF-02: Gestión de perfil	50
20 REQF-02: Gestión de perfil, Actualizar usuarios	51
21 REQF-02: Gestión de perfil, depuración de los datos enviados mediante la petición POST	52
22 REQF-02: Gestión de perfil, Mensaje de verificación para eliminar algún usuario	52
23 REQF-02: Gestión de perfil	53
24 REQF-05: Gestión de documentos, Mensaje de alerta	54
25 REQF-07: Notificaciones, alerta exitosa	54
26 REQF-09: Almacenar bitácora de los usuarios, salida de un procedimiento.	55
27 Formato inicial de los archivos JSON.	61
28 Tabla ROLES	62
29 Tabla USUARIOS	62
30 Tabla REGISTRO	63
31 Tabla HISTORIAL CURRICULAR	63
32 Tabla RESOLUCIONES	63
33 Tabla ASIGNATURA HISTORIAL	63

RESUMEN

“La Universidad Austral de Chile es una institución acreditada que forma profesionales y graduados de pre y postgrado, con un sello caracterizado por la excelencia académica, el compromiso con la libertad y con el medio sociocultural, el respeto por la diversidad, la responsabilidad social, entre otros” [Alt7]. El cumplimiento de estas definiciones establecidas en el *modelo educacional y enfoque curricular*, requieren, entre otros, de procesos internos de la organización que apoyen la gestión educativa, en particular, de pregrado. Una de las funcionalidades más importantes en este ámbito, es la gestión de los proyectos curriculares de las carreras. Por esto es de gran importancia el conocer el historial curricular de cada carrera, necesidad que da origen al presente proyecto.

El objetivo principal del presente proyecto de tesis consiste en diseñar y desarrollar un prototipo de plataforma web que permita gestionar el historial curricular de cada carrera de la Universidad Austral de Chile, el cual permitirá a distintas unidades de la universidad tener una mejor información curricular de las carreras y así facilitar el trabajo que día a día realizan.

El sistema web se desarrollará para las dependencias de la Universidad Austral de Chile, es por eso mismo que el alumno tesista debe adaptarse a las tecnologías que la universidad utiliza, por esta razón la solución se desarrollará en las siguiente tecnologías: Microsoft Visual Studio 2013, Microsoft SQL Server 2008 (solo en ambiente de desarrollo, una vez finalizado el proyecto se migrará a Sybase, el cual es el motor de base de datos que utiliza la universidad), Visual Basic como lenguaje del servidor, JavaScript, CSS3, HTML5 y GitHub como gestor de versiones.

El sistema web beneficiará a los departamentos del área de pregrado de la Universidad que requieren permanentemente trabajar con información curricular de las carreras, estos departamentos son los siguientes: Departamento de Aseguramiento de la Calidad e Innovación Curricular (DACIC), Departamento de Registro Académico Estudiantil y Departamento de Admisión y Matricula, además beneficiará a la propia escuela, ya que les permitirá contar con información histórica curricular.

Una de las principales ventajas obtenidas por esta plataforma, es la de mejora los tiempos de respuesta que requieren los distintos departamentos al momento de necesitar algún dato

estadístico referente a los cambios que se le efectúan a las mallas curriculares.

ABSTRACT

“ Universidad Austral de Chile, is an accredited institution that educates professionals and undergraduate and postgraduate students with a signature that is characterized by its academic excellence, commitment to freedom, sociocultural environment, respect for diversity, social responsibility, among others” [Alt7]. The accomplishment of these definitions established in the educational model and curricular approach, requires among others, of internal processes of the organization, to support the educational management, in particular of the undergraduate studies. One of the most important functions in this field, is the management of the curricular designs of the careers. Therefore, it is very important to recognise the curricular history of each career, a necessity that gives its origin to the present project.

The main objective of the present dissertation project, consists in designing and developing a platform web prototype that allows to manage the curricular story of each Universidad Austral de Chile’s career, which will allow all different units of the university to have a better curricular information of the careers and facilitate the work they do every day.

This web system will be developed for Universidad Austral de Chile, therefore the dissertation student should adapt himself to the technologies that the university utilises, reason why the solution will be developed in the following technologies: Microsoft Visual Studio 2013, Microsoft SQL Server 2008 (only for development, once the project is completed, it will migrate to Sybase, which is the engine of the database of the University), Visual Basic and as language of the server, JavaScript, CSS3, HTML5 and GitHub as a server manager.

This web system will benefit all undergraduate departments of the University that permanently require to work with the curricular information of the careers, which are the following ones: Quality and Curricular Innovation Assurance Department (DADIC), Student Academy Register Department and Admission and Enrolment Department, it will also benefit the faculty, because it will allow to have with all historical curricular information.

One of the main advantages of this platform, is the improvement in the response times that the different departments require at the moment of the need of any piece of statistical data

referring to the changes in the curriculum.

1. INTRODUCCIÓN

"Durante la última década, las universidades pertenecientes al Consejo de Rectores de Universidades chilenas (CRUCH) han promovido diversas iniciativas de Innovación Curricular" [Tum11]. Es por ello que la Universidad Austral de Chile ya ha empezado con el proceso de innovación curricular para que gradualmente abarque todas sus carreras.

Uno de los principales problemas relacionados con el proceso de calidad e innovación curricular es que la Universidad Austral de Chile almacena toda la información referente al historial curricular de cada carrera en distintos medios de almacenamiento (incluyendo el papel), distintos formatos, y en varias unidades de la organización, lo que dificulta la generación de informes que apoyen procesos estratégicos de seguimiento y auto-evaluación.

Este proyecto surge como una iniciativa de la Dirección de Estudios de Pregrado, donde se observan las dificultades que tienen los departamentos ya indicados previamente, al momento de gestionar documentos relacionados con la innovación curricular. De acuerdo a lo mencionado anteriormente, se pretende crear una plataforma web, que permita gestionar el historial curricular de cada carrera de la universidad, la cual permitirá a distintas Unidades de la universidad tener una mejor información de las carreras y así facilitar el trabajo que día a día realizan.

La información obtenida por esta plataforma servirá de apoyo para la gestión curricular por parte de los distintos departamentos que integran la Dirección de Estudios de Pregrado, ya que se contará con todo lo necesario para gestionar los cambios curriculares de los planes de estudio.

1.1. Definición del Proyecto

1.1.1. Objetivo general

Diseñar y construir un prototipo de una plataforma web que apoye al monitoreo curricular de pregrado.

1.1.2. Objetivos específicos

- Conocer cómo los departamentos que integran la Dirección de Estudios de Pregrado (DEP) administran la información referente a los planes de estudios.
- Definir requerimientos del sistema, describiendo sus funcionalidades y separar en módulos la aplicación.
- Diseñar e implementar el módulo necesario que permita gestionar el historial curricular de una carrera en particular.
- Diseñar e implementar el módulo necesario que permita la gestión de documentos.
- Realizar pruebas de validación de los requisitos y estabilidad del prototipo de plataforma web.

1.2. Nivel actual

Como ya se ha mencionado el presente proyecto pretende aportar a los procesos curriculares de la Universidad Austral de Chile, es por esto mismo que es necesario entender los procesos que realizan los departamentos involucrados.

En la Figura 1 se puede apreciar como se estructuran los departamentos de Vicerrectoría. A partir de este diagrama se explicarán las principales funciones y procesos de los Departamentos de Estudios de Pregrado, los cuales son: Departamento de Aseguramiento de la Calidad de la Docencia e Innovación Curricular, Departamento de Admisión y Matricula y Departamento de Registro Académico Estudiantil.

1.2.1. Departamento de Aseguramiento de la Calidad e Innovación Curricular (DACIC)

En este departamento nace la idea de desarrollar un sistema que apoye a los procesos curriculares. Su principal objetivo es “propender al fortalecimiento de la calidad de los aprendizajes de los estudiantes de la UACH a través del apoyo a los Docentes en el diseño de situaciones de enseñanza/aprendizaje orientadas a la obtención de resultados efectivos y el asesoramiento a las unidades académicas respectivas en el desarrollo de innovaciones curriculares.” [Dac15]

¹Elaboración propia.

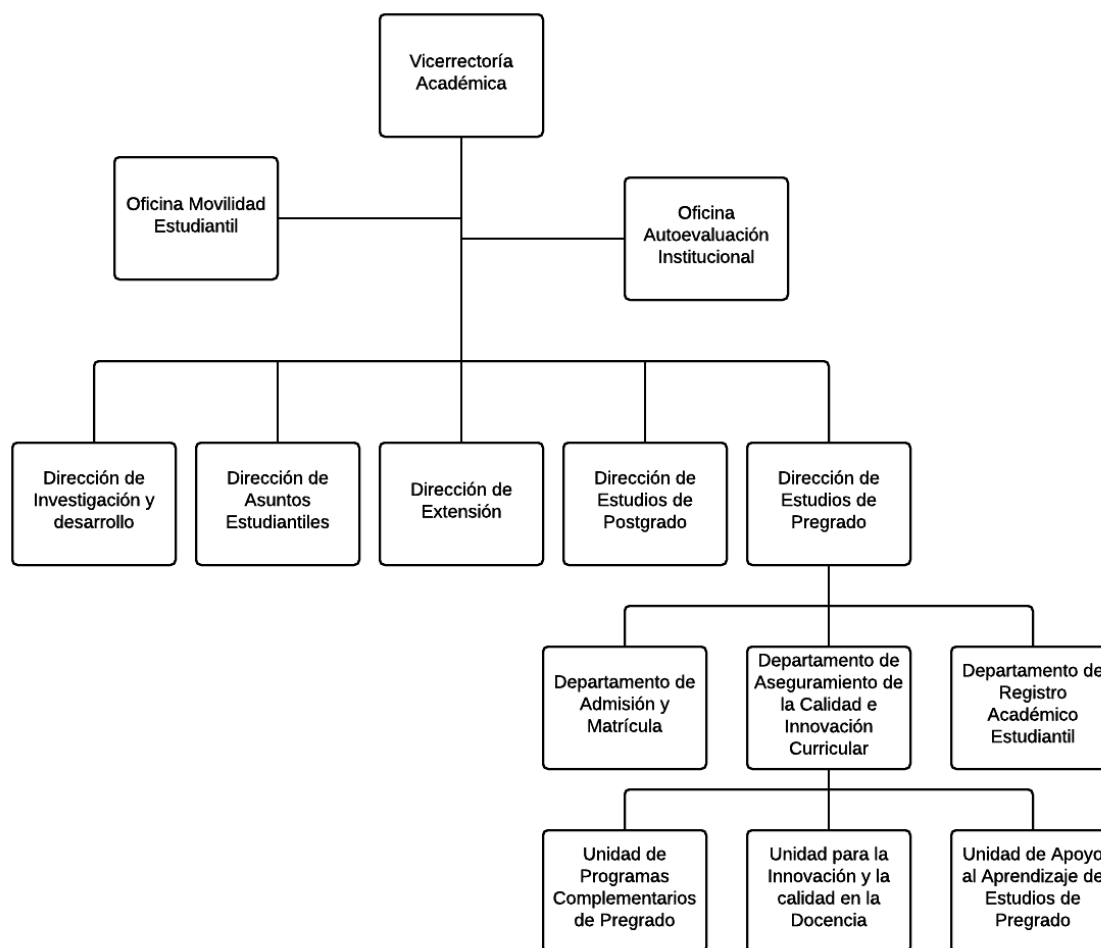


Figura 1. Organigrama Vicerrectoría y DACIC ¹

El departamento está conformado por cuatro Unidades:

- Unidad de Apoyo al Desarrollo de la Docencia de Pregrado
- Unidad de Apoyo al Desarrollo y la Innovación Curricular en Pregrado
- Unidad de Apoyo al Aprendizaje del Estudiante de Pregrado
- Unidad de Programas Complementarios

Sin embargo el proyecto beneficiará directamente a la Unidad de Apoyo al Desarrollo y la Innovación Curricular en Pregrado, por lo que se dará mas detalles de esta Unidad a continuación.

1.2.1.1. Apoyo al Desarrollo y la Innovación Curricular en Pregrado

Unidad encargada de apoyar en el ámbito “técnico-curricular a las escuelas de Pregrado en sus proyectos de innovación curricular, tanto en carreras profesionales como técnicas, en el contexto del Modelo Educativo y Enfoque Curricular de la Universidad." [Dac15]

1.2.2. Departamento de Admisión y Matrícula

Es la Unidad encargada del proceso de ingreso de los alumnos vía PSU a cada una de las carreras de Pregrado que ofrece la universidad Austral de Chile. Es decir, es la unidad que está presente en la gestión de los procesos de ingresos a la Universidad.

Existen dos modalidades para ingresar a la Universidad Austral de Chile, las cuales son:

- Sistema regular, común a toda la Educación Superior adscrita al Consejo de Rectores de las Universidades Chilenas.
- Sistema de Ingreso Especial, propio de la Universidad.

1.2.2.1. Servicios estudiantiles

- “ El Departamento de Admisión y Matrícula es la Unidad encargada de otorgar las certificaciones de Alumno Regular a todos los estudiantes de la Universidad, que cumplan con los requisitos para la de emisión de dichos documentos.” [[Dep15](#)]
- Gestiona la construcción y entrega de las Credenciales Universitarias.

1.2.3. Departamento de Registro Académico Estudiantil

Este departamento depende directamente de la Dirección de Estudios de Pregrado de la Vicerrectoría Académica, está actualmente a cargo de la Srta. María Cristina Barriga Ramírez. Entre sus funciones se destaca “el seguimiento académico del estudiante de pregrado, postítulo y postgrado desde su primera matrícula hasta su egreso y posterior titulación y/o graduación y el permanente apoyo a la labor administrativa que realizan Directores de Escuela, Directores de Unidades Académicas y Profesores en general” [[Dir15](#)].

1.2.3.1. Procesos

Esta Unidad controla la información académica - administrativa, prepara y supervisa los siguientes procesos:

- Peticiones de asignaturas que realizan las Escuelas.

- Oferta de asignaturas de las Unidades Académicas.
- Inscripción de asignaturas de los estudiantes.
- Ingreso de las calificaciones por parte de los profesores.
- Modificaciones curriculares de los planes de estudios, de acuerdo a lo aprobado por la Dirección de Estudios de Pregrado.

Una vez nombrado los diferentes procesos que controla este departamento, se procederá a explicar los dos principales procesos que están directamente relacionado con el sistema, los cuales son: Creación de nuevas Carreras y Modificaciones curriculares de los planes de estudios, de acuerdo a lo aprobado por la Dirección de Estudios de Pregrado.

1.2.3.2. Creación de nuevas Carreras

En el diagrama de procesos que se muestra en la Figura 2 describe la secuencia básica que se lleva a cabo para la creación de nuevas carreras en la Universidad Austral de Chile.

Las fechas para la presentación de nuevos proyectos o innovaciones curriculares, se estipulan en el Decreto de Rectoría que promulga el Calendario Académico cada año. Para el año 2015 se especifica:

- 30 de abril, último día para presentar en la Vicerrectoría Académica, los proyectos de nuevas carreras.
- 30 de junio, último día para que las facultades y sedes presenten proyectos de Innovación Curricular de las carreras a la Vicerrectoría Académica.

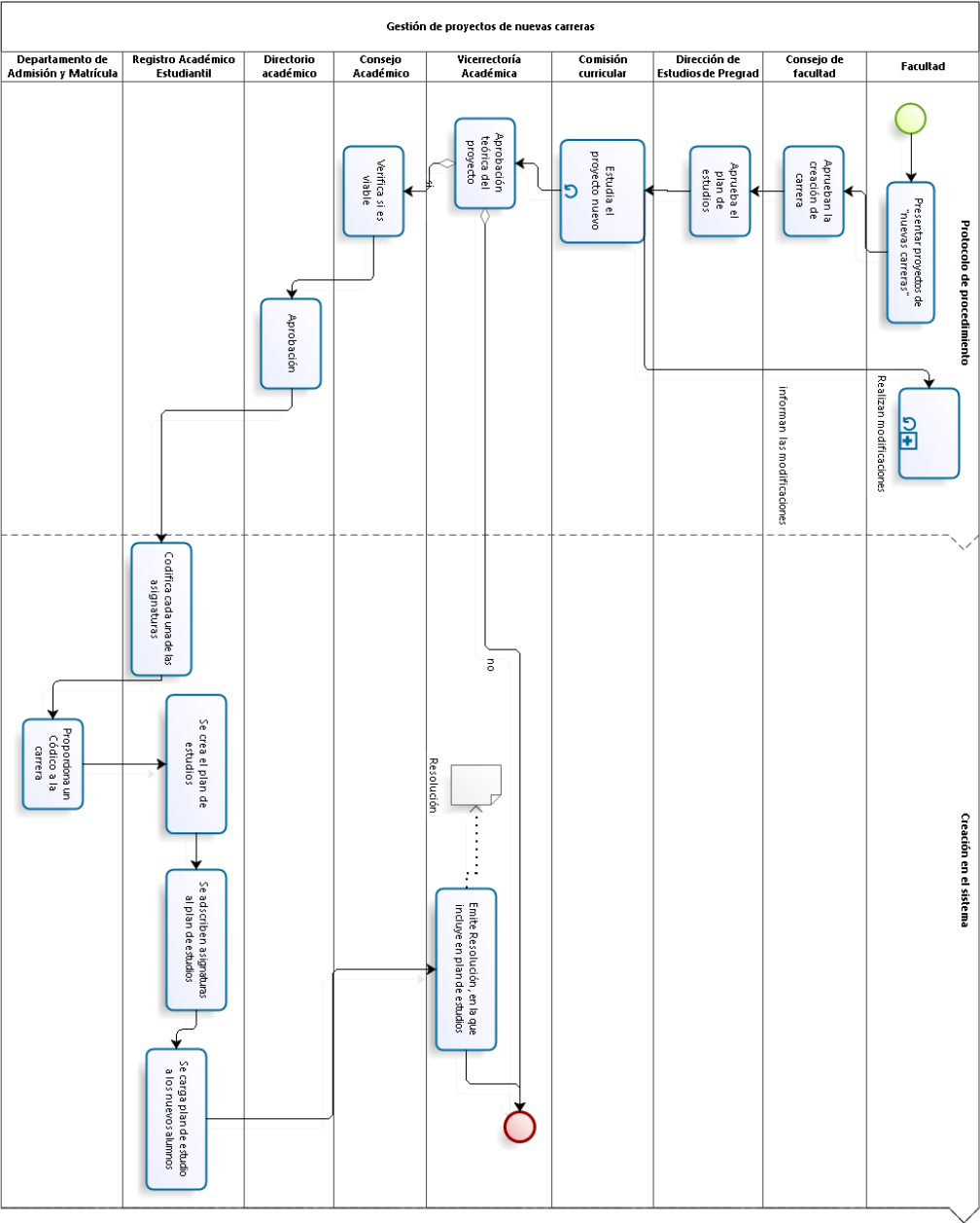


Figura 2. Proceso de presentación de nuevos proyectos ²

Una vez aprobado una nueva carrera o la innovación curricular por parte de la Vicerrectoría Académica, los pasos administrativos, son los que se indican a continuación:

1. Se codifica cada una de las asignaturas, asociándose a una Unidad Académica específica. La Unidad Académica es la que le asigna la sigla, en letras y la numeración es un correlativo que se utiliza para la identificación de la asignatura. Ejemplo: CAEV222-14.
CAEV = Ciencias Ambientales y evolutivas (Unidad Académica)
222 = Numeración asignada para identificarla dentro de la misma unidad.
-14 = año de creación (2014)

²Elaboración propia.

2. El Departamento de Admisión y Matrícula proporciona un código a la carrera.
Ejemplo 1826, Escuela de Psicología (valdivia).
3. Una vez que las carreras estén creadas y las asignaturas asociadas a una unidad académica, el Departamento de Registro Académico Estudiantil crea el plan de estudios (ver Figura 3) con la descripción del propio plan:Nombre de bachillerato,Duración, Año de creación,e etc.
4. El proceso final para la creación de una carrera, es cargar las asignaturas al plan previamente creado, los pasos son los siguientes:
 - a) Se ingresa una a una las asignaturas previamente creadas.
 - b) Se asocia a un semestre en particular.

1.2.3.3. Modificaciones Curriculares Mayores y/o Menores

Tanto la creación de nuevas carreras como todo lo que tiene que ver con modificaciones con respecto al plan de estudios, son proceso que ve el DACIC, específicamente el sub-departamento de Unidad de Apoyo al Desarrollo y la Innovación Curricular en Pregrado, en conjunto con Registro Académico Estudiantil.

La Figura 3 muestra los procesos administrativos que realiza Registro Académico Estudiantil, a continuación se explicará con mayor detalle. El proceso de modificación del plan de estudios comienza con la iniciativa de una escuela, la cual se reúne previamente con Registro Académico Estudiantil, con el objetivo de identificar quiénes son los afectados por los cambios (número de alumnos, año de ingreso, plan de estudios, etc.). Una vez concretada esta reunión, la escuela se encuentra en condiciones de formalizar la petición, la cual es enviada a Dirección de Estudios Pregrado con la intención de que sea evaluada. En caso de que la petición sea viable, Dirección de Estudios de Pregrado envía una comunicación interna a Registra Académico Estudiantil, informándole que puede efectuar los cambios.

Por último Registro Académico Estudiantil realiza una petición de requerimientos a la DTI (en caso de que sea necesario, de lo contrario no se efectúa este proceso) para posteriormente aplicar los cambios e informarle a la escuela si Dirección de Estudios Pregrado rechazó o aprobó las modificaciones curriculares.

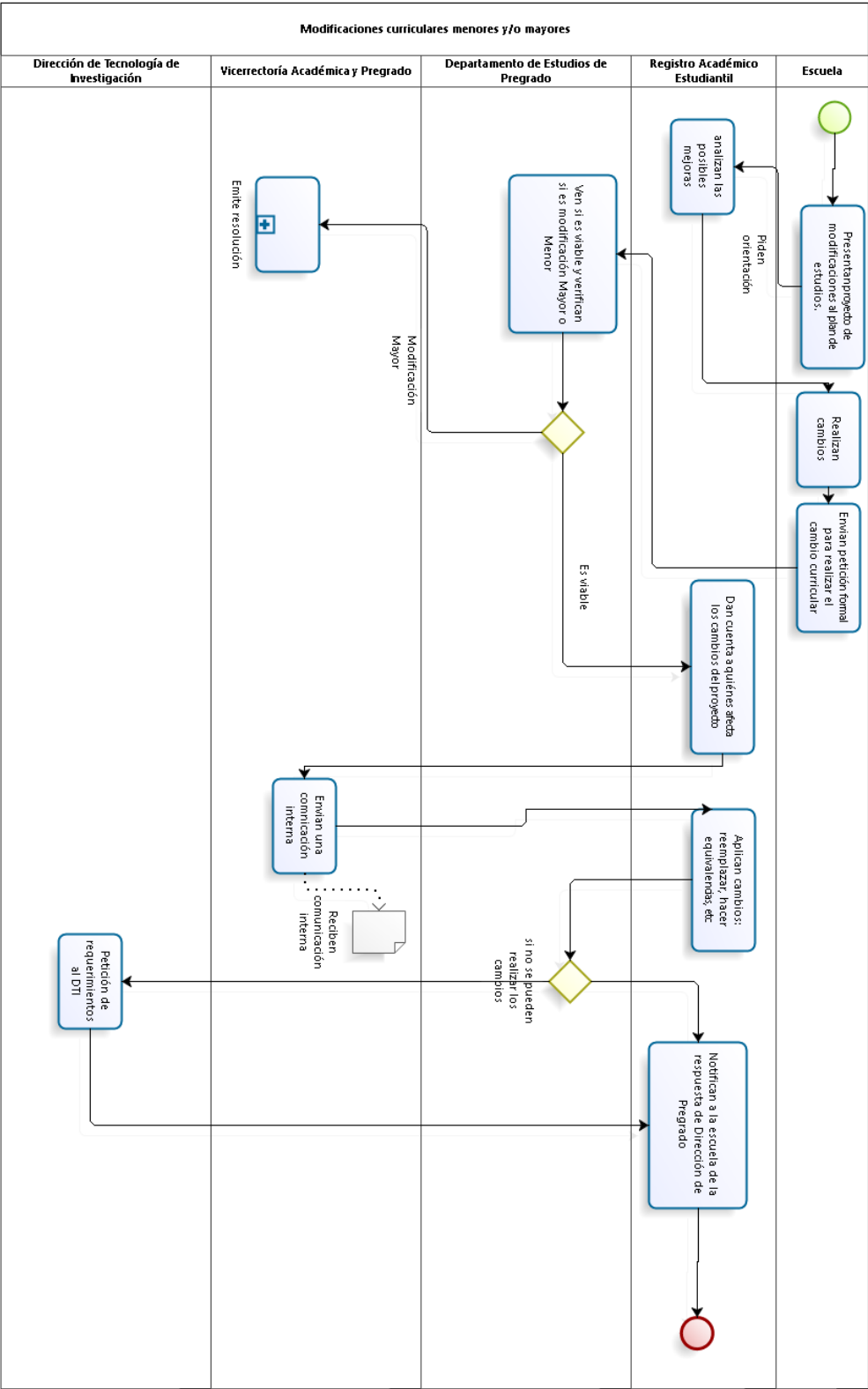


Figura 3. Procesos de cambios Curriculares Mayores y/o menores ³

³Elaboración propia.

Dado lo anterior, podemos ver que los procesos curriculares no se gestionan con un sistema informático, no existiendo un software encargado de almacenar y documentar los cambios que afectan a los proyectos curriculares de las carreras.

1.3. Metodología

Para cumplir con los objetivos de este proyecto, es necesario adquirir conocimientos de los procesos administrativos de los departamentos involucrados en el proyecto, lo cual permitirá entender cómo estas oficinas manipulan la información referente a los planes de estudio.

Para conocer los procesos mencionados anteriormente, se debe trabajar en conjunto con el DACIC, el cual proporcionará toda la información bibliográfica relacionada con los cambios de las mallas curriculares, es decir; resoluciones, diagrama de procesos, formularios de programas innovados, etc.

Para finalizar la etapa teórica, se investigó sobre las tecnologías en las cual se desarrollará el software, que como ya se ha mencionado anteriormente, las tecnologías asociadas al proyecto tienen que ser compatibles con las que usa la Universidad Austral de Chile.

La siguiente etapa del proyecto, es diseñar y desarrollar el software. Para esta etapa se definió una metodología de entrega incremental, ya que en un principio los requisitos no estaban muy claros al comienzo del proyecto. El desarrollo se dividió en tres incrementos. En el primer incremento se seleccionaron los requisitos que se relacionaban con la gestión del historial curricular de una carrera en particular, el segundo incremento se seleccionan los requisitos referentes a la gestión de documentos, por último, el tercer incremento corresponde a la Autenticación y gestión de perfiles de usuarios.

Para finalizar, se realizó una etapa de validación del software, donde se probó la funcionalidad y usabilidad de éste, con el fin de comprobar si cumplía con los requisitos propuestos. Para validar las funcionalidades esperadas del software y la usabilidad del mismo, se trabajo con la Sra. Pilar Alarcón, secretaria de Registro Académico Estudiantil, quien utilizó el software y ayudó a poblar el sistema desarrollado.

2. MARCO TEÓRICO

En esta sección se exploran los elementos que intervienen en el desarrollo del software. Se hace un recorrido desde las tecnologías que se utilizarán en el desarrollo, hasta la descripción de los documentos que utilizarán los usuarios del proyecto.

2.1. Arquitectura

En este capítulo se presenta la arquitectura y las tecnologías involucradas en el desarrollo de la plataforma. Debido a que en los requisitos del proyecto se contempla que la plataforma quede operativa en la DTI⁴ de la Universidad, el proyecto se tuvo que adaptar a la arquitectura con la que trabaja este departamento.

2.1.1. Pseudo MVC

El patrón de arquitectura MVC⁵ es una filosofía de diseño de aplicaciones que define la organización independiente del Modelo, la Vista y el Controlador [Eje15]. De esta forma, el sistema se divide en tres capas donde el **modelo** contiene una representación de los datos que maneja el sistema, es decir, el modelo de datos, la **vista** o interfaz de usuario contiene la información que se envía al cliente y los mecanismos de interacción con éste, y por ultimo, el **controlador** es el intermediario entre el modelo y la vista, gestionando el flujo de información entre ellos.

Como se ha dicho la arquitectura MVC utiliza la división tradicional en 3 capas (presentación, lógica de negocio y datos) sin embargo la arquitectura de los proyectos de la Universidad no es específicamente MVC, es más bien una adaptación puesto que no se separa totalmente lo que es vista del controlador, pero si se manejan de forma separada la capa de datos de la capa lógica y la interfaz de usuario, de manera semejante es como trabaja ASP.NET (se explicará en la sección 2.2.2.2) . La arquitectura descrita anteriormente se detallará a continuación.

⁴Dirección de Tecnologías e Información de la Universidad Austral de Chile

⁵Modelo-vista-Controlador

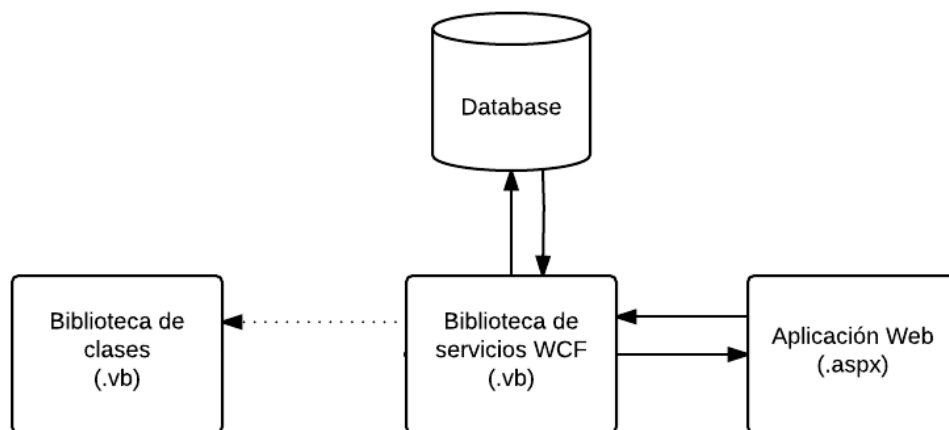


Figura 4. Arquitectura pseudo MVC Universidad Austral de Chile ⁶

En la Figura 4 se puede apreciar las 3 capas de esta arquitectura. Esta arquitectura esta compuesta por:

1. **Biblioteca de servicios WCF:** Esta capa responde a eventos, usualmente acciones del usuario, e invoca los procedimientos almacenados del modelo de datos, el cual obtiene la información en la interfaz genérica de Visual Basic: IEnumerable.
2. **Biblioteca de clases:** Esta capa es la encargada de darle formato a la información obtenida por la biblioteca de servicios WCF, es decir, es la representación específica de la información con la cual el sistema opera.
3. **Aplicación Web:** Esta capa es la encargada de interactuar con el usuario, en ella se muestra toda la información ordenada y entendible por el usuario, sin embargo no solo posee código front-end (html, css, JavaScript, etc), si no también métodos que se comunican con los servicios WCF. Es por ello que la vista no esta del todo separada de las reglas de negocio.

Si bien estas tres capas son importantes, la capa que se explicará a continuación es la primera, es decir, la biblioteca de servicios WCF, puesto que las bibliotecas de clase no son mas que un conjunto de clases que definen las entidades de la base de datos, y la aplicación web es el conjunto de archivos(aspx, js, css, jpg, etc) que permiten al usuario realizar acciones sobre el Sitio Web.

⁶Elaboración propia.

2.1.1.1. Biblioteca de servicios WCF

Hasta el momento, el servicio de mensajería entre aplicaciones se realizaba mediante los protocolos COM, DCOM o MSQM, que obligaba a los programadores a ceñirse no sólo a una forma de programación concreta, sino que también estaba atada a la plataforma y al lenguaje de programación. Los servicios web surgen con el propósito de cambiar esta filosofía, permitiendo hacer la comunicación independiente de lenguaje de programación y plataforma gracias a la creación de estándares de comunicación [Gar13].

Hoy en día existen diversos protocolos sobre los que los servicios web pueden operar, sin embargo son dos los protocolos mas usados por los servicios web:

- **SOAP:** *Simple Object Access Protocol*. Utiliza XML como lenguaje de codificación, y su principal ventaja es que puede ser transmitido por cualquier capa de transporte (HTTP, TCP/IP, SMTP, etc).
- **REST:** *Representational State Transfer*. A diferencia del protocolo anterior, REST solo puede utilizar la capa de transporte HTTP, sin embargo su filosofía se basa en la ausencia de estado y la “equivalencia” entre los cuatro verbos que pueden utilizarse en el protocolo HTTP y las cuatro operaciones CRUD (Create, Read, Update, Delete) básicas que pueden realizarse sobre una fuente de datos.

Windows Communication Foundation (WCF) es un marco de trabajo para la creación de aplicaciones orientadas a servicios, su principal objetivo es el de unificar las comunicaciones, es decir las aplicaciones que utilicen WCF pueden implementar varios protocolos de servicio web, por lo que permite que el código sea independiente del protocolo que se utilice.

WCF logra su independencia mediante la separación entre operaciones y datos, en otras palabras, un servicio web WCF establece un contrato a través de una interfaz y una clase es la encargada de implementarla. De este modo, un servicio WCF, se compondrá de los siguientes componentes:

- **Contrato de servicio (ServiceContract):** expone una operación que el servicio web es capaz de ejecutar. Corresponde a una interfaz.
- **Contrato de datos (DataContract):** implementa un tipo de dato que el servicio

web será capaz de manejar. Generalmente, será el tipo de dato que manejará el contrato de servicio.

- **Implementación del servicio:** implementará la interfaz correspondiente al contrato de servicio, haciendo uso del contrato de datos para intercambiar la información.

Los anteriores componentes de un servicio WCF se pueden apreciar gráficamente en la Figura 5.

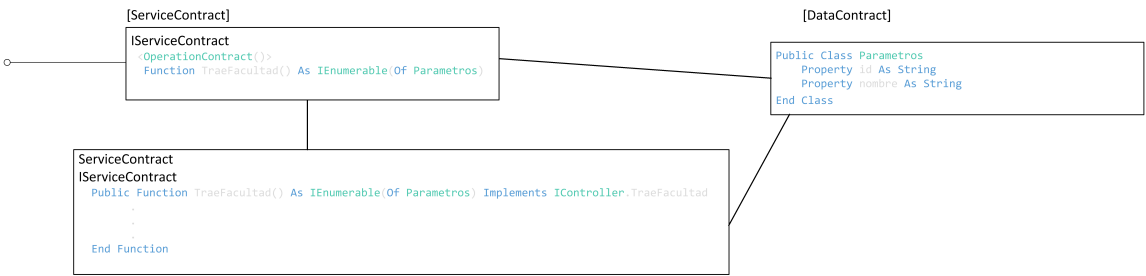


Figura 5. Diagrama de componentes de un servicio WCF ⁷

Podemos condensar lo dicho hasta aquí, que el framework que utiliza la Universidad Austral de Chile es una forma distinta de crear aplicaciones web, si bien, se identifican claramente la división tradicional en 3 capas (presentación, lógica de negocio y datos) no es la misma que propone el patrón MVC. En particular, en esta arquitectura la capa **Controller** no tendría una clara correspondencia en la estructura clásica, si no que es una mezcla de la interfaz gráfica y la lógica del negocio.

2.2. Tecnologías para el desarrollo de la plataforma

A continuación se presentan las distintas tecnologías utilizadas para el desarrollo de la plataforma en sus respectivas capas.

2.2.1. Front-end

Esta capa es la parte del software que interactúa con el o los usuarios, en ésta se encuentran todas las tecnologías que corren del lado del cliente, es decir, todas aquellas tecnologías que corren del lado del navegador web. Es por ello que es de vital importancia de que el front-end sea capaz de entregar al usuario todas las herramientas necesarias para que éste pueda realizar una correcta interacción con el sistema. Entre las tecnologías usadas en esta capa se encuentran las habituales en el desarrollo Web, tales como HTML, CSS y JavaScript junto con otras que se describirán a continuación.

⁷Elaboración propia.

2.2.1.1. JQuery

JQuery es una librería JavaScript rápida, liviana y con amplias funcionalidades. Hace mucho más simple tareas como recorrer y manipular un documento HTML, manejar eventos, animaciones, e interacciones Ajax ⁸ por medio de una API fácil de usar que funciona a través de múltiples navegadores. Con una combinación de versatilidad y capacidad de ampliación, esta librería busca cambiar la forma en que las personas escriben JavaScript [JQu15].

Si bien existen otras librerías de JavaScript (Prototype, MoonTools, entre otros). Se decidió usar JQuery en el proyecto por los siguientes motivos:

- Es de uso general, por lo que posee una comunidad activa y una extensa documentación.
- Posee una amplia variedad de complementos que facilitan el desarrollo.
- Es modular.
- Es compatible con todos los navegadores existentes.

2.2.1.2. Alertify

Alertify es un script escrito con JQuery, el cual nos permite utilizar los siguientes elementos Javascript personalizados: alert(), confirm() y prompt(). Además también nos permite utilizar sus notificaciones, las cuales son muy agradables y sencillas de utilizar y modificar [Ale15].

Alertify ha sido construido para personalizar nuestras alertas y notificaciones, de esta manera el front-end de la plataforma es mas amigable al usuario y esto permite un mejor entendimiento de los eventos que se realizan en tiempo de ejecución. Además es un plugin multi-idioma y posee responsive-design ⁹

A pesar de que la mayoría de los plugins de notificaciones presentan inconvenientes al momento de implementar con vb.net, se decidió usar un sistema de alertas principalmente

⁸Ajax: Asynchronous JavaScript And XML

⁹ **Responsive Design** es un nuevo paradigma del desarrollo web. Permite adaptar cada sitio a los diferentes formatos de dispositivos de acceso; smartphones, tabletas, portátiles, etc.

para facilitar la visualización de todos los eventos que ocurren en el sistema, y Alertify fue el plugin que menos problemas presentó al momento de la integración con vb.NET.

2.2.1.3. Bootstrap

Bootstrap fue creado a mediados del 2010 por un diseñador y un desarrollador de la red social Twitter, es un proyecto de código abierto, y es uno de los frameworks de front-end más populares en el mundo. Sirvió como guía de estilo para el desarrollo de herramientas internas en la empresa durante más de un año antes de su lanzamiento público, y continua haciéndolo hoy en día [Boo15].

El uso de un framework hace posible que el desarrollo del front-end sea: Fácil, ya que la mayoría de los framework posee una curva de aprendizaje baja, es decir, poseen una gran eficiencia en el aprendizaje, lo que permite dominar la mayoría de los componentes en un tiempo reducido; Optimizado para dispositivos móviles, puesto que bootstrap posee todas las reglas CSS necesarias para hacer que los sitios se adapten dinámicamente a la gran mayoría de pantallas y resoluciones existentes en el mercado.

2.2.1.4. Template Metis

Un template es un conjunto de archivos que determinan la estructura y el aspecto visual de un sitio web, y tiene como ventaja principal disminuir tiempos y costos de desarrollo [Jof15].

El uso de un template disminuye el tiempo de desarrollo de un diseño web, por lo que el alumno tesista se puede centrar en la funcionalidad del sistema que es lo principal. "Metis es una template de administración gratuito basado en Twitter Bootstrap 3.x" [Git15], fue diseñado por un usuario de gitHub y lo puso a disposición para que cualquier usuario lo pueda utilizar.

2.2.1.5. Parsley

Parsley es una librería de validación de formularios, le ayuda a proporcionar a los usuarios información sobre su envío de formulario antes de enviarlo al servidor [Par15]. Fue creado

por Guillaume Potier y actualmente es sustentado por Marc-André.

El uso de esta librería se sustenta en el hecho de que notificar los errores de forma inmediata mejora la satisfacción del usuario con la aplicación y ayuda a reducir la carga de procesamiento innecesario en el servidor.

2.2.2. Back-end

El back-end es el área que se dedica a la parte lógica de un sitio web, es el encargado de que todo funcione como debería, el back-end no es visible para el usuario ya que no se trata de diseño, o elementos gráficos, se trata de programar las funciones que tendrá un sitio.

2.2.2.1. Microsoft SQL Server 2008

Microsoft SQL Server 2008 Express es un sistema de administración de datos eficaz y confiable que ofrece un variado conjunto de características, protección de datos y rendimiento para clientes de aplicaciones incrustadas, aplicaciones web ligeras y almacenes de datos locales [[Mic15b](#)].

Microsoft Sql Server esta catalogado como una herramienta fácil de instalar, usar y administrar, por lo que muchas organizaciones hacen uso de sus servicios, tales como Itaú, Samsung, Marcopolo, entre otras grandes compañías.

Para un manejo adecuado de la base de datos se utilizará SQL Server Management Studio(SSMS), que es “ un entorno integrado para obtener acceso, configurar, administrar y desarrollar todos los componentes de SQL Server.SSMS combina un amplio grupo de herramientas gráficas con una serie de editores de script enriquecidos que permiten a desarrolladores y administradores de todos los niveles obtener acceso SQL Server"[[Mic15c](#)].

2.2.2.2. ASP.NET

ASP.NET es una plataforma web que proporciona todos los servicios necesarios para compilar aplicaciones web empresariales basadas en servidor. ASP.NET está compilado

en .NET Framework, por lo que todas las características de .NET Framework están disponibles en las aplicaciones ASP.NET. Las aplicaciones se pueden escribir en cualquier lenguaje que sea compatible con Common Language Runtime (CLR), incluido Visual Basic y C# [Mic15a].

ASP.NET introduce el concepto del code-behind en el desarrollo web, el cual separa el código de la interfaz de usuario, es decir, todo lo relacionado con la Interfaz de usuario se maneja en el archivo .aspx y el control de los eventos en un archivo separado .vb (Visual Basic). Con ello se facilita la programación de aplicaciones en múltiples capas, lo que en definitiva se traduce en la total separación entre lo que el usuario ve y lo que la base de datos tiene almacenado. La ventaja del modelo code-behind es que no se lee secuencialmente sino que se compila, lo que incrementa la velocidad de respuesta del servidor. Además, al compilarse, el incremento en seguridad y fortaleza es muy grande.

Otro aspecto a tener en cuenta de ASP.NET es que sirve tanto para Webs sencillas como para grandes aplicaciones, ya que la orientación a objetos y la naturaleza compilada permite que la programación de sitios web sea sencilla.

2.2.3. Herramientas Anexas

Esta sección contiene información detallada acerca de todas las herramientas que facilitaron la creación e implementación del proyecto. No se clasifican ni en Front-End ni en Back-End dado que son herramientas de gestión y de corrección de errores, por lo que el software final no las contendrá.

2.2.3.1. GitHub

El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante [GitH15], dicho de otra manera, un control de versiones permite guardar “fotografías” del estado de un proyecto en ese instante del tiempo, dando la capacidad de restaurar ese estado en cualquier momento. Esto ayuda a trabajar en un proyecto y si sale algo mal se puede volver atrás, a algún punto en donde todo funcionaba correctamente.

A pesar de que existen diversos sistemas de control de versiones y diversos métodos de control de versiones (control de versiones locales, control de versiones centralizados y control de versiones distribuidos), se escogió GitHub por las siguientes razones.

1. Sistema distribuido: en el que todos los nodos manejan la información en su totalidad y por lo tanto pueden actuar de cliente o servidor en cualquier momento, es decir, se elimina el concepto de “centralizado”.
2. Trabajo en local: Al tener una copia del proyecto se pueden hacer commits de forma local, una vez que se tenga conexión a internet se realiza el commits al servidor.
3. Fotografías, no diferencias: Cuando un archivo no cambia, en lugar de guardar la misma “fotografía” varias veces, guarda una referencia a esa “fotografía”. De esta forma se optimizan los recursos del sistema.

Es necesario recalcar que en el proyecto de titulación solo existe un programador, por lo que no es necesario que el gestor de versiones sea centralizado, de manera que GitHub cumple con lo que se necesita para el desarrollo del software, el cual es guardar las versiones en caso de cometer algún error en el futuro.

2.2.3.2. Firebug

Muchas veces es difícil saber qué errores se cometen en el desarrollo y cómo solucionarlos, pero con ayuda de algunos programas, se puede reducir el tiempo que se pierde en la búsqueda de los problemas o errores cometidos.

Si bien uno de los requisitos no funcionales es que la aplicación funcione en los 3 navegadores mas usado a nivel mundial, el desarrollo y las pruebas se hicieron bajo el navegador Firefox, por lo que se tenia que tener alguna herramienta compatible con este navegador para poder encontrar los errores de manera fácil.

Firebug es un plugin de Firefox que nos brinda un paquete de utilidades para el desarrollo de páginas y aplicaciones Web. Nos permite debugear, monitorizar y modificar el CSS, HTML y JavaScript en tiempo de ejecución [Moz15]. Dicho de otra manera, Firebug no solo debuguea los errores, si no que también el usuario puede monitoriar todas las peticiones de red que se realizan, entre las mas utilizadas: Petición GET, Petición POST.

2.3. Tipos de documentos

Como se mencionó en la Sección 1, existen distintos tipos de documentos que dan origen a alguna modificación curricular, los cuales se nombrarán a continuación.

- Comunicación Interna.
- Correos electrónicos.
- Resolución (Vicerrectoría Académica, Dirección de Estudios de Pregrado, Secretaria General).
- Decretos.
- Programas de asignaturas.

3. DESARROLLO DEL SISTEMA

En este capítulo no solo se detallan los procesos de ingeniería de software; Planificación, Análisis, Diseño e implementación y Validación, sino también se presentan los principales artefactos UML.

3.1. Planificación

El objetivo de esta sección es describir el marco de trabajo en el cual se desarrolla el software, es decir, se especifica la metodología de desarrollo utilizada y se muestra el plan de trabajo.

3.1.1. Metodología

Todo desarrollo de software es riesgoso y difícil de controlar, pero si no se lleva a cabo una metodología de por medio, se obtiene clientes insatisfechos con el resultado y desarrolladores aun mas.

Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información [Tin10]. Una metodología esta formada por fases, cada una de las cuales se puede dividir en sub-fases. Las fases que la mayoría de las metodologías poseen son:

1. Análisis y definición de requerimientos.
2. Diseño de la arquitectura.
3. Desarrollo del sistema software.
4. Validación del sistema.

En pocas palabras las metodologías ayudan a tener una buena administración y gestión sistemática de todo proyecto de software, y llevarlo a cabo con altas posibilidades de éxito. De esta forma es posible crear, desarrollar y mantener un sistema desde que surge la necesidad del producto hasta que se cumple el objetivo por el cual fue desarrollado.

El ciclo de vida utilizado para el desarrollo de este proyecto es el de entrega incremental. Este ciclo de vida entrega el software en partes pequeñas, pero utilizables, llamadas

incrementos. El primero incremento es un producto esencial, en otras palabras, se afrontan requisitos básicos, pero muchas funciones suplementarias quedan sin extraer. El cliente utiliza el incremento con el fin de que identifiquen cuales son los servicios más y menos importantes, por consiguiente, se definen varios incrementos en donde cada uno proporciona un subconjunto de funcionalidades del sistema. Este proceso se repite siguiendo la entrega de cada incremento, hasta que se finalice el proyecto completo.

3.1.2. Plan de trabajo

En la Figura 6 se presenta el plan de trabajo para llevar a cabo la implementación del proyecto, siguiendo las etapas de desarrollo de la ingeniería de software de acuerdo al modelo de entrega incremental.

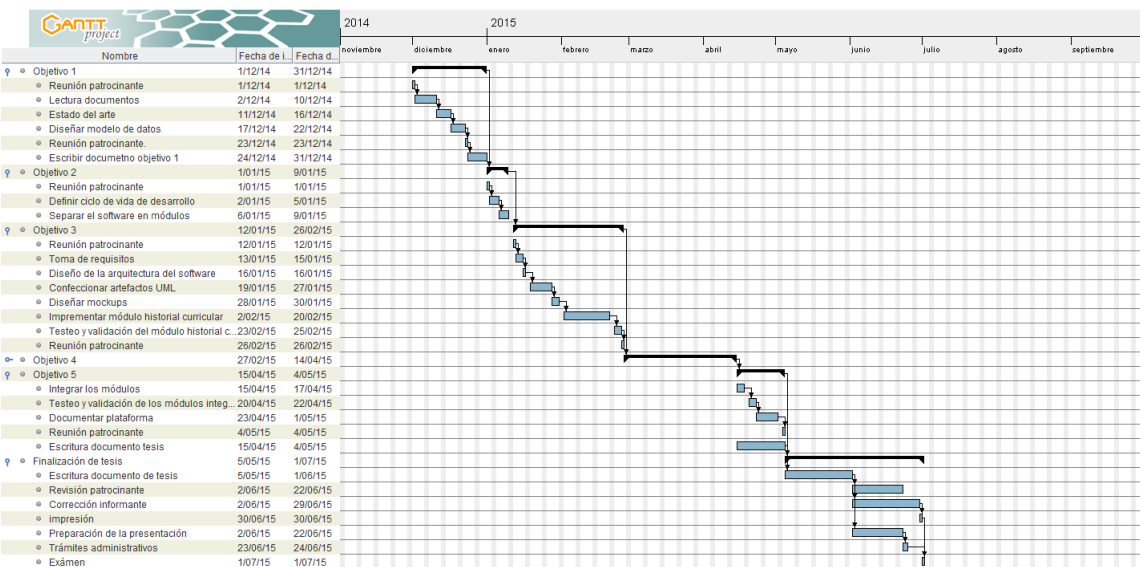


Figura 6. Carta Gantt que exhibe las etapas de desarrollo del sistema ¹⁰

Durante el proceso de desarrollo del producto software es importante tener en cuenta lo siguiente:

- Es importante reunirse con el profesor patrocinante al comienzo, durante y al término de una iteración, para así aclarar y resolver requerimientos específicos del sistema que puedan surgir.
- El producto final se dará terminado cuando se hayan cumplido todos los requisitos del sistema.

¹⁰Elaboración propia.

3.2. Análisis

En esta sección se describen los requerimientos y casos de uso más representativos de cada módulo, los cuales fueron obtenidos y discutidos mediante reuniones con el profesor patrocinador, con el objetivo de identificar los requisitos funcionales y no funcionales que deberían satisfacer el producto software final.

3.2.1. Requerimientos

A continuación se presenta una lista de los requisitos que se definieron para el producto software.

3.2.1.1. Requisitos funcionales

Los requisitos funcionales del sistema se presentan en la Tabla 1.

Tabla 1. Requisitos funcionales	
REQF-01	Autenticación de usuario
Descripción	El sistema debe ser capaz de diferenciar distintos tipos de usuarios, mediante el RUT, tipo de usuario y contraseña. Estos pueden ser de tres tipo: Administrador, Editor, y Subscriptor.
REQF-02	Gestión de perfil
Descripción	El sistema debe permitir que los usuarios puedan modificar su contraseña. Además debe permitir que el Administrador pueda modificar los datos que el usuario ingresó al momento de registrarse.
REQF-03	Desplegar historial curricular
Descripción	El sistema debe permitir que todos los usuarios puedan ver el conjunto de hitos curriculares de una carrera en particular, mediante la facultad, la escuela y la carrera previamente ingresados por el usuario.

Continúa en la página siguiente

Tabla 1 – *Continuación de la pagina anterior*

REQF-04	Registro de usuarios
Descripción	El sistema deberá permitir el registro de nuevos usuarios, los cuales deben ser creados por el Administrador. Los datos a solicitar son: RUT, nombre, apellido paterno, apellido materno, correo electrónico y rol. La contraseña será los 6 primeros dígitos del RUT ingresado.
REQF-05	Gestión de documentos
Descripción	El sistema debe permitir al Administrador y Editor crear nuevos registros curriculares, indicando el plan de estudio, la carrera y el tipo de hito: Modificación mayor, Modificación menor o Innovación Curricular, además debe permitir subir uno o mas archivos en formato PDF por registros, los cuales el sistema los tienen que clasificar en: Resolución, Comunicación Interna y Petición de requisitos.
REQF-06	Visor de PDF
Descripción	El sistema debe permitir al usuario ver cualquier documento que se encuentre almacenado en la base de datos.
REQF-07	Notificaciones
Descripción	El sistema desplegará distintos tipos de notificaciones al momento de realizar cualquier tipo de cambio (guardar, editar, eliminar). Los dos tipos de alertas que se consideraron en la plataforma web son : Success y Error .
REQF-08	Almacenar bitácora de los usuarios
Descripción	El sistema internamente debe almacenar todas las notificaciones de los eventos ocurridos en la plataforma a fin de tener un registro de todas las actividades que realizan los usuarios en la plataforma. Un registro de la bitácora debe tener necesariamente esta compuesto por: Código de la alerta, mensaje de la alerta, fecha y el RUT de usuario quien ejecutó dicha alerta.
REQF-09	Visualización de bitácora
Descripción	El sistema debe permitir al administrador visualizar todos los eventos ocurridos en el sistema.

3.2.1.2. Requisitos no funcionales

Los requisitos no funcionales se presentan en la Tabla 2.

Tabla 2. Requisitos no funcionales

REQNF-01	Ambiente Web
Descripción	El sistema debe visualizarse y funcionar correctamente en cualquier navegador, especialmente en Internet Explorer, Mozilla y Google Chrome.
REQNF-02	Escalabilidad
Descripción	El sistema debe estar en capacidad de permitir en el futuro el desarrollo de nuevas funcionalidades, modificar o eliminar funcionalidades.
REQNF-03	Facilidad de uso
Descripción	La interfaz del sistema debe ser amigable con el usuario, Mensajes de errores y éxito.
REQNF-04	Facilidad de pruebas
Descripción	El sistema debe contar con facilidad para la identificación de la localización de los errores durante la etapa de prueba.
REQNF-05	Validación
Descripción	El sistema tiene que poseer una interfaz en la cual se validen los campos obligatorios y manejo de datos que se ingresan.
REQNF-06	Acoplación
Descripción	El sistema tiene que estar programado con la estructura que utiliza la DTI, esto es para facilitar el acoplamiento una vez finalizado el proyecto

3.2.2. Modelo conceptual

Como apoyo al análisis de requisitos, en la Figura 7 se muestra a continuación un modelo conceptual tomando en cuenta todos los requisitos, de esta manera se tiene una visión global del dominio del problema a abordar, y de los distintos conceptos presentes en él y de cómo se relacionan.

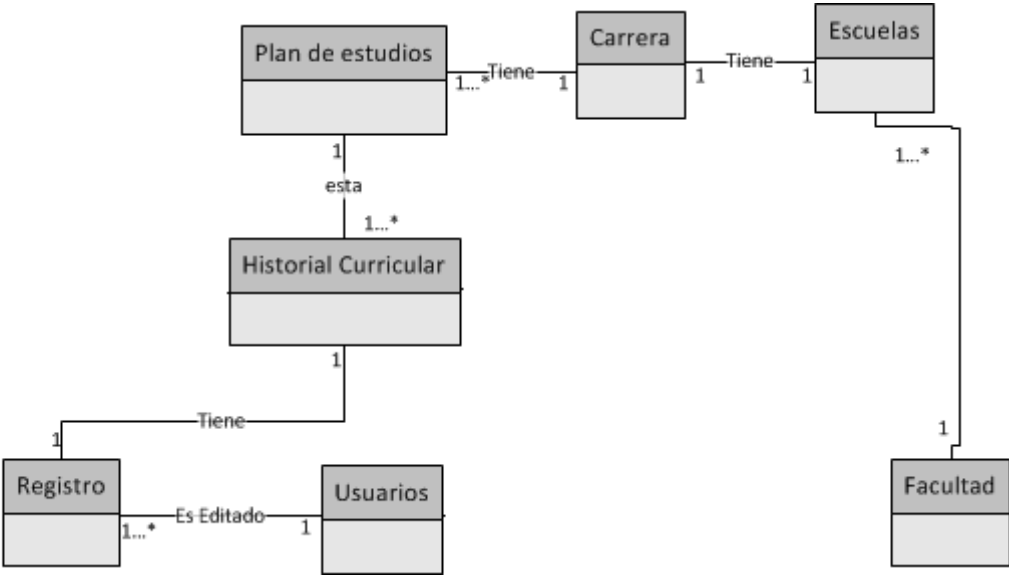


Figura 7. Modelo conceptual del proyecto ¹¹

3.2.3. Casos de uso

En las Figuras 8, 9, 10 se presentan los diferentes casos de uso para los actores involucrados en el sistema.

¹¹Elaboración propia.

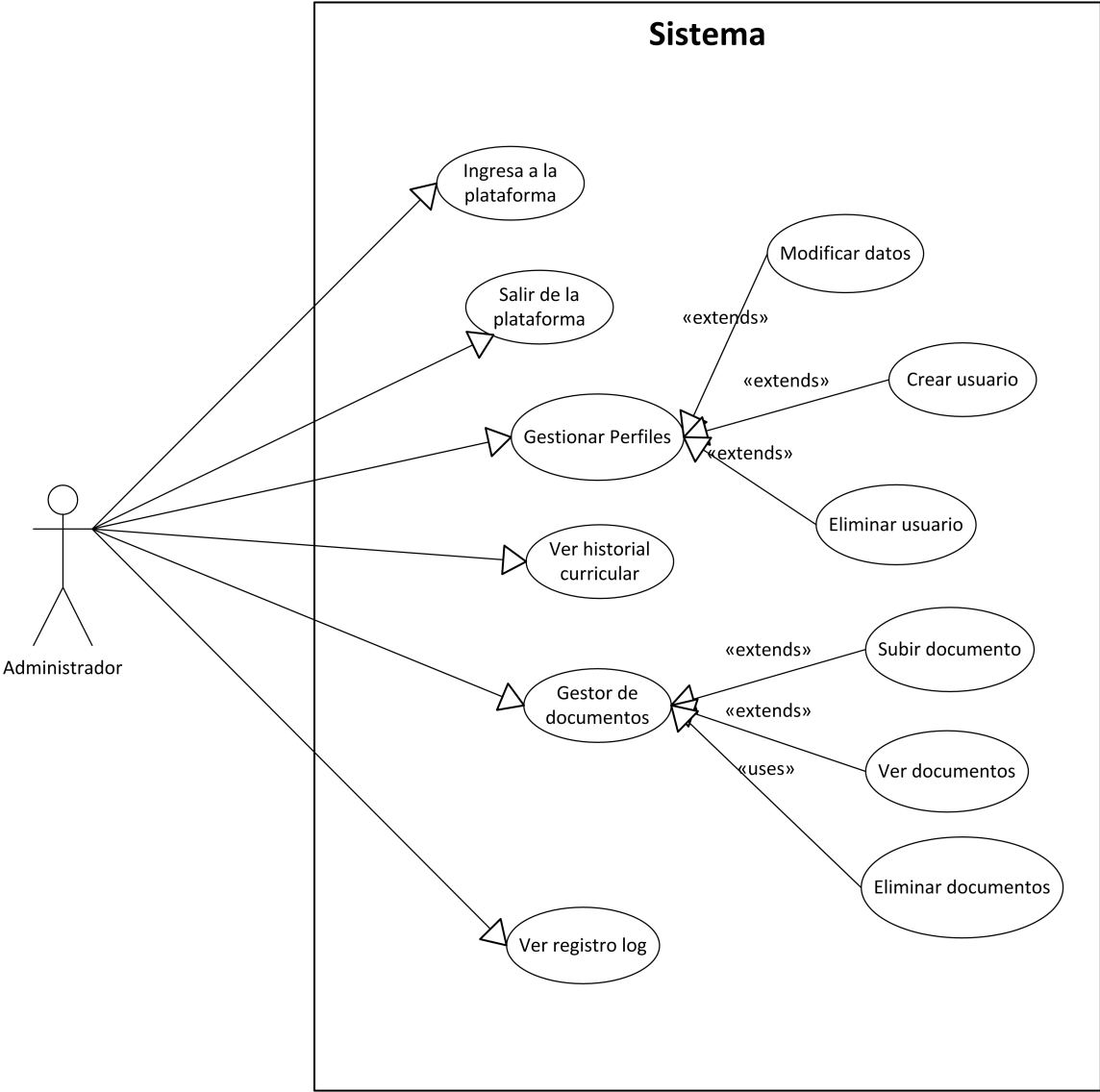


Figura 8. Diagrama de caso de uso para el Administrador ¹²

¹²Elaboración propia.

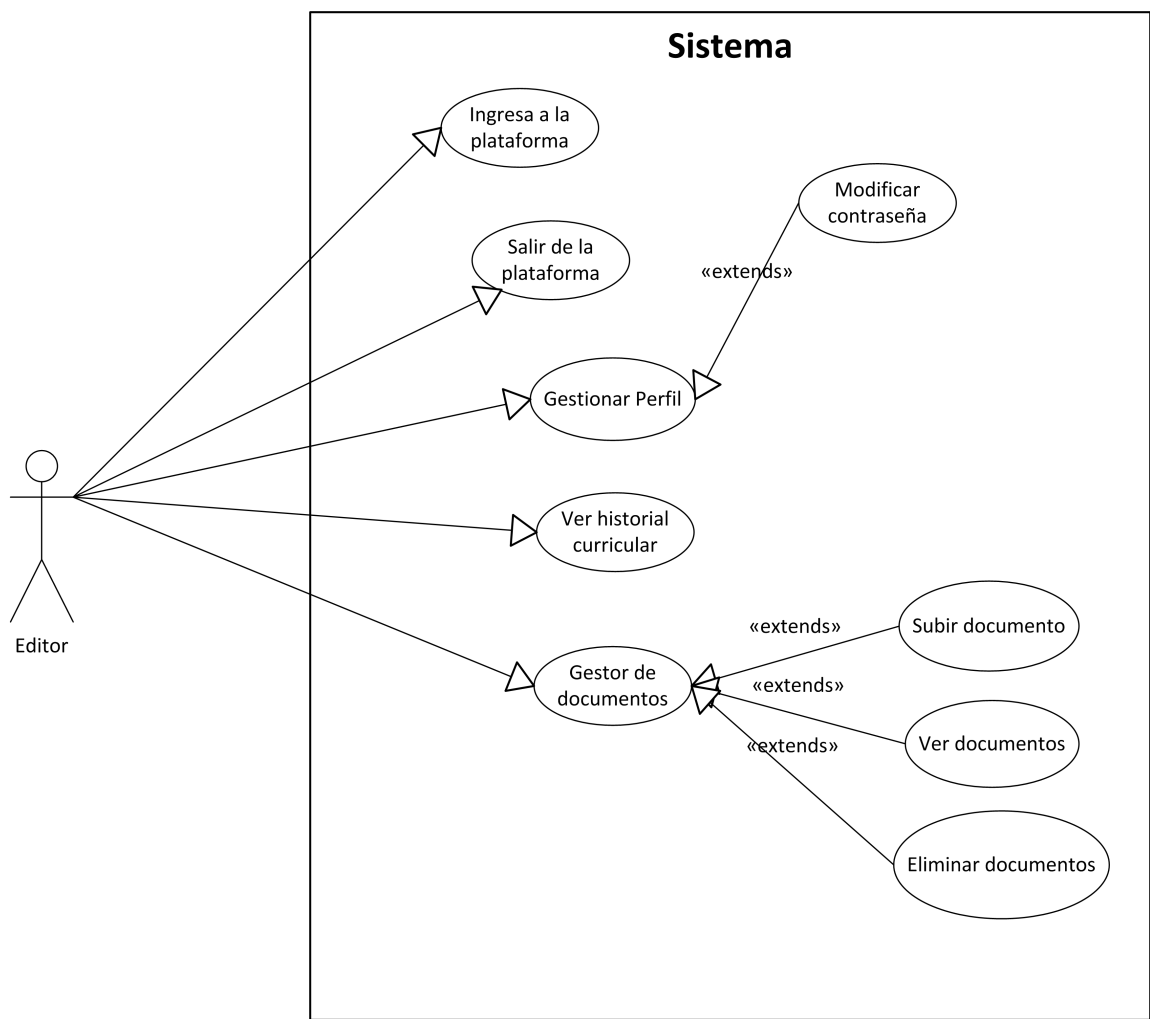


Figura 9. Diagrama de caso de uso para el Editor¹³

¹³Elaboración propia.

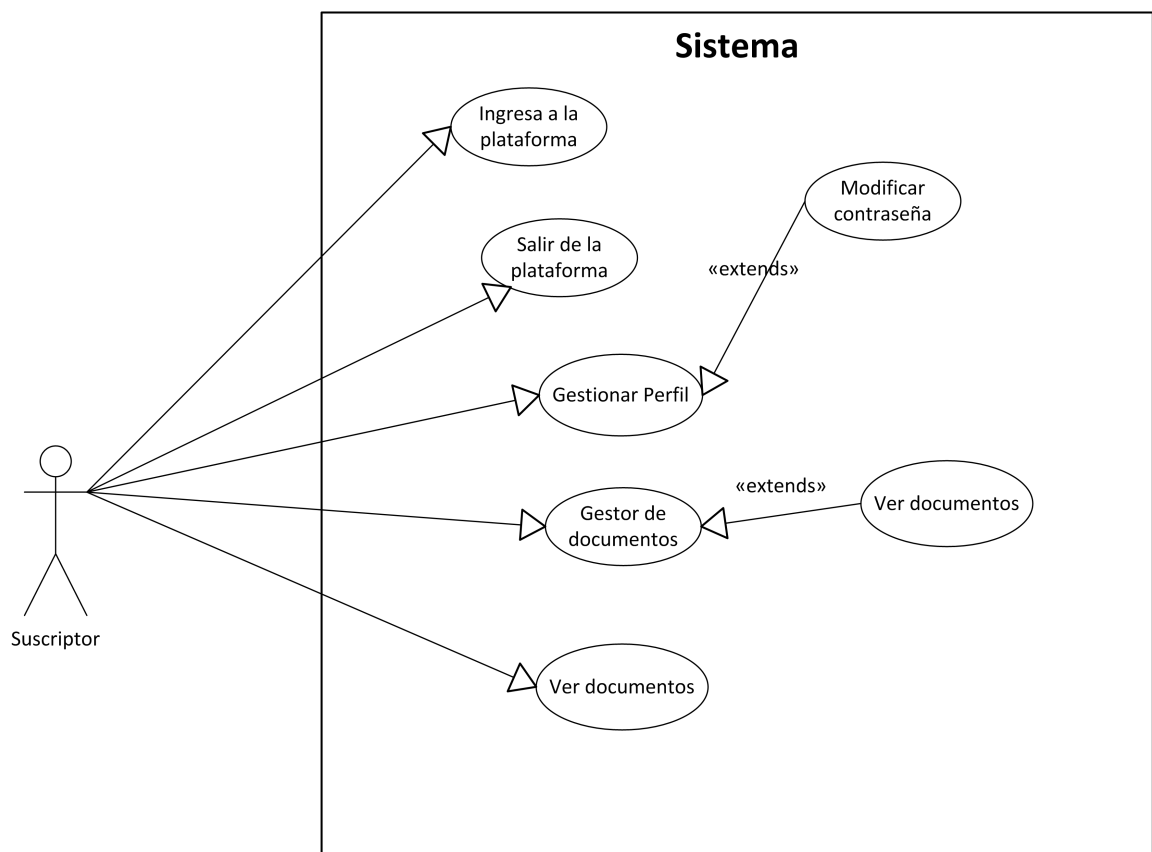


Figura 10. Diagrama de caso de uso para el Suscriptor ¹⁴

3.2.3.1. Descripción de los actores del sistema

Suscriptor: corresponde al grupo de individuos que pueden revisar los datos en la plataforma, es decir, pueden ver el historial curricular de una carrera y ver todas las resoluciones. sin embargo no posee ningún permiso de edición o creación.

Editor: Este grupo de individuos posee todos los permisos necesarios para poder realizar todas las operaciones básicas (crear, leer, editar y borrar) sobre los registros curriculares y/o los documentos que están en el sistema.

Administrador: Corresponde al individuo o grupo de individuos que tiene todas las funciones de un creador de registros curriculares, pero que además es el encargado de administrar la totalidad de usuarios de la plataforma, lo cual incluye el registro, edición y eliminación de usuarios.

¹⁴Elaboración propia.

3.3. Diseño

En esta sección se describe el diseño y la implementación para los módulos: *Historial curricular* y *Gestor de documentos* de la plataforma. En un principio se muestran algunos artefactos generales correspondientes al diagrama de componentes y al modelo de datos. Posteriormente para cada módulo se presentaran los casos de uso, además del diagrama de secuencia correspondiente.

3.3.1. Diagramas de componentes

En la Figura 11 y 12 se muestran los diagramas de componentes software y hardware respectivamente. Ambos diagramas son congruentes con lo visto en el Capítulo 2.1.1, correspondiente al patrón de diseño de programación que utiliza ASP.NET.

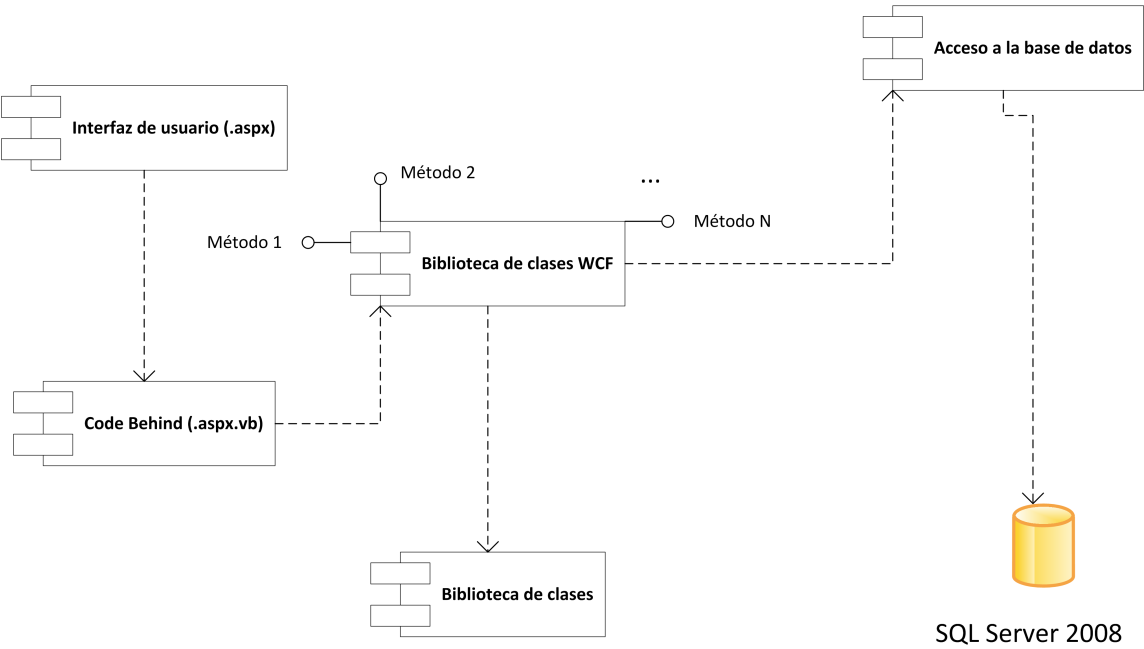


Figura 11. Diagrama de componentes Software del sistema¹⁵

- **Interfaz de usuario:** La interfaz de usuario consta de los archivos aspx, los cuales no solo definen la estructura de las páginas en el sistema sino también es el encargado de llamar todos los archivos CSS y JS necesarios para el correcto funcionamiento.
- **Code Behind:** Como se mencionó en la Sección 2.2.2.2, ASP.NET permite organizar los eventos en forma separada de la interfaz. Todo lo relacionado con

¹⁵Elaboración propia.

- Interfaz de usuario se maneja en el archivo .aspx y el control de los eventos en un archivo separado .aspx.vb.
- **biblioteca de clases WCF:** Este componente contiene los servicios web de la aplicación.
 - **biblioteca de clases:** Conjunto de clases que definen las estructuras de las entidades de la base de datos.
 - **Acceso a la base de datos:** El acceso a la base de datos se realiza mediante un DLL que fue facilitado por la DTI.

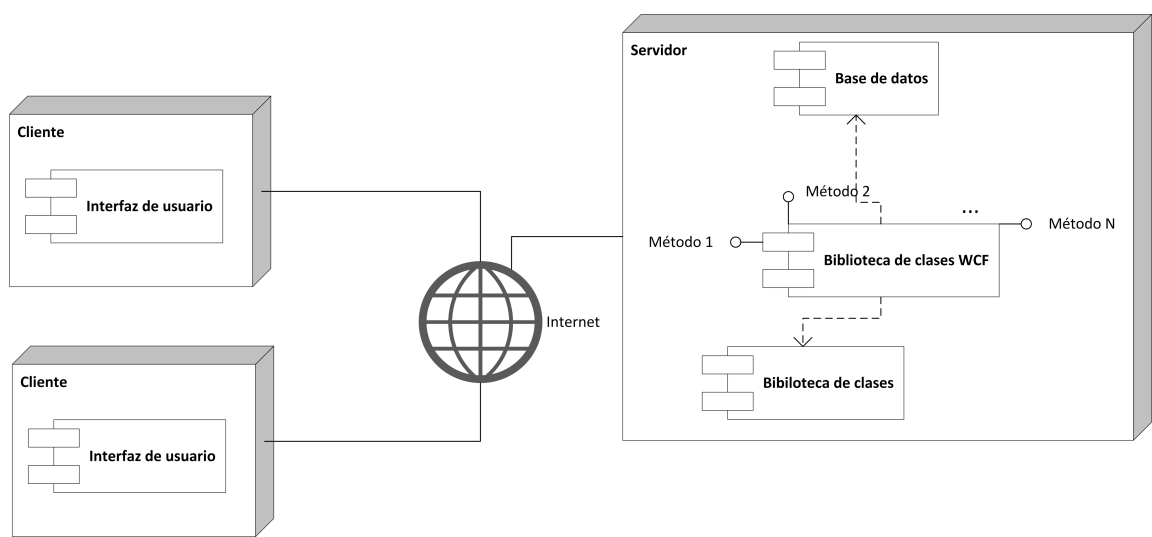


Figura 12. Diagrama de componentes Hardware del sistema¹⁶

3.3.2. Modelo de datos

El contexto en el que se desenvuelve el software contempla 14 entidades, de las cuales 8 de ellas se exportaron desde la base de datos de la Universidad para poder trabajar en un ambiente de desarrollo, estas entidades son: Facultad, Escuelas, Carreras, PlanEstudio, PlanAsignatura, Plan_asig_requisito,institutos y asignaturas.

Las entidades restantes fueron creadas con el fin de satisfacer las necesidades de la problemática en la cual se encuentra inmerso en proyecto. El modelo de datos completo se puede ver en la Figura 13.

¹⁶Elaboración propia.

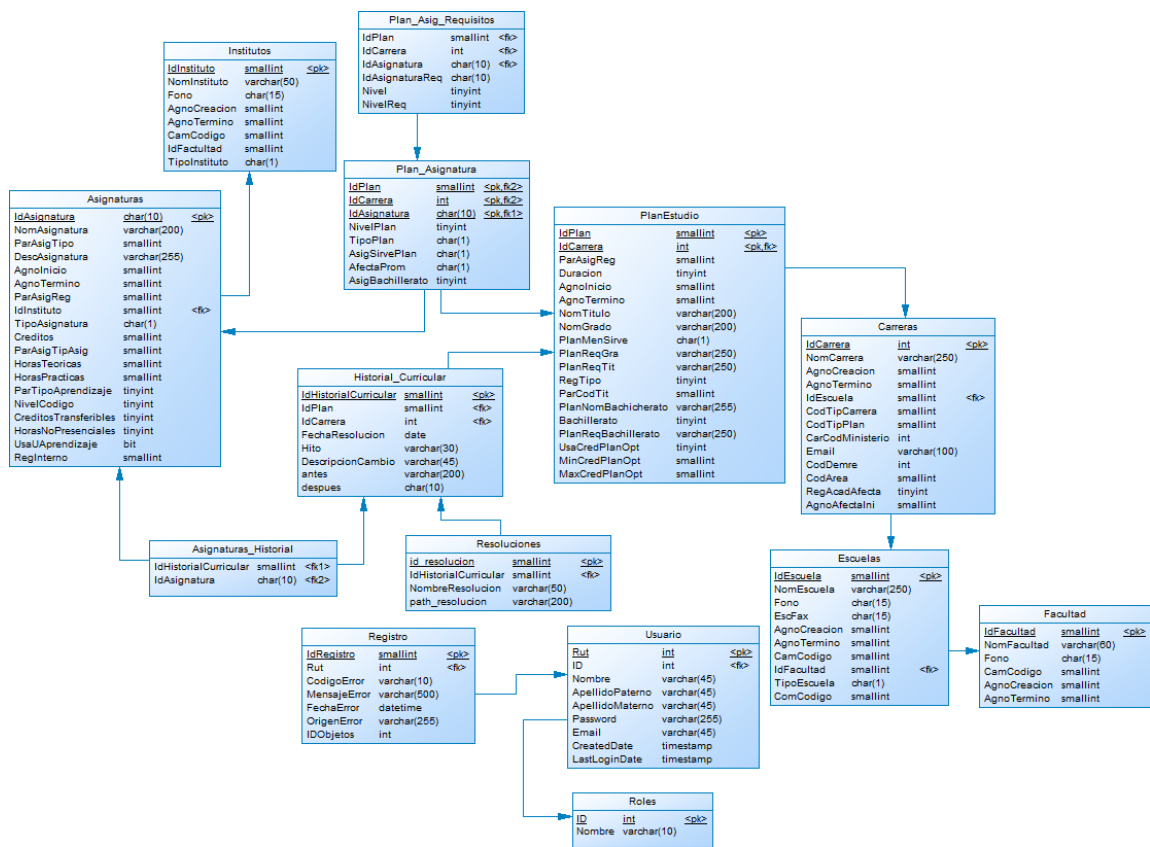


Figura 13. Modelo de datos Entidad-Relación ¹⁷

3.3.2.1. Diccionario de datos

Es importante que todo proyecto el cual involucre un almacenamiento de datos, posea un diccionario de datos, puesto que el objetivo de un diccionario de datos es dar precisión sobre los datos que se manejan en un sistema, evitando así malas interpretaciones.

Como se mencionó anteriormente, no todas las entidades son de creación propia, por lo que el en Anexo C se puede ver el diccionario de datos correspondiente solo a las tablas creadas.

3.3.3. Módulo historial curricular

El módulo historial curricular esta diseñado para que todos los usuarios definidos en la Sección 3.2.3.1 tengan acceso a él, a causa de que su principal objetivo es ver la trazabilidad de una carrera en particular.

¹⁷Elaboración propia.

3.3.3.1. Diseño

A continuación se describe el diseño del caso de uso más significativo para el módulo historial curricular de la plataforma, que corresponde a la visualización de la trazabilidad de los planes de estudio de las carreras . Se presentará en detalle la composición de este caso de uso y su participación en el sistema.

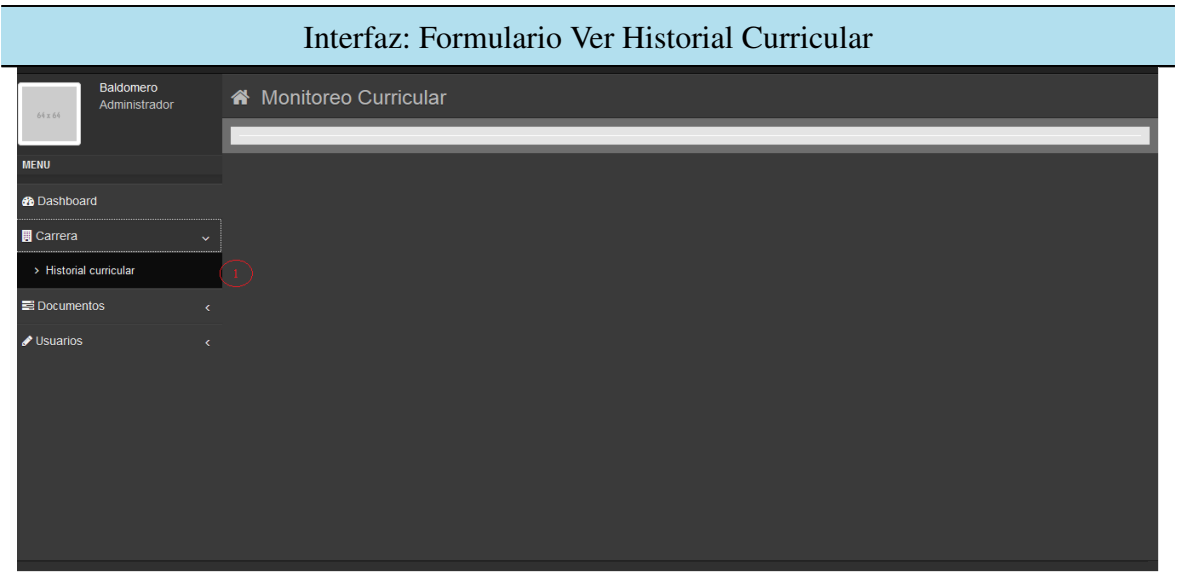
Caso de uso real más significativo

El caso de uso “Ver historial curricular” ocurre cuando un usuario desea ver los cambios curriculares que se le han aplicado a una carrera en particular. El caso de uso real “Ver historial curricular” se presenta en la Tabla 3.

Tabla 3. Caso de uso Ver historial curricular

Caso de Uso Ver Historial Curricular	
Actores	Administrador, Editor, Suscriptor.
Propósito	Visualizar los cambios curriculares que se han realizado en una determinada carrera.
Tipo	Primario y esencial

Resumen: Un usuario autenticado puede ver el historial curricular de una carrera en particular, el cual lo hace primero seleccionado la facultad a la que pertenece, luego la escuela y por último selecciona la carrera en cuestión.



Continúa en la página siguiente

Tabla 3 – Continuación de la página anterior

Monitoreo Curricular

A

Historial Curricular

Seleccione facultad

Seleccione Facultad

Seleccione Escuela

Seleccione Escuela

Seleccione carrera

Seleccione Carrera

Detalle Historial Curricular

Mostrar 10 registros

Año

Fecha

Hito

Descripción

Antes

Después

Ningún dato disponible en esta tabla

Año

Fecha

Hito

Descripción

Antes

Después

Mostrando registros del 0 al 0 de un total de 0 registros

Anterior

Siguiente

Monitoreo Curricular

Historial Curricular

Seleccione facultad

Facultad de Ciencias Agraria

Seleccione Escuela

Agronomía

Seleccione carrera

Agronomía

Detalle Historial Curricular

Mostrar 10 registros

Año

Fecha

Hito

Descripción

Antes

S

2014

28-05-2015

Modificación mayor

Cambio de instituto que dicta la carrera con código CAEV-143

CAEV-143, instituto de ciencias a

B

2015

30-10-2015

Modificalcón menor

Agregar requisito en el tercer nivel de la asignatura ICML035

ICML035 no posee requisitos.

Año

Fecha

Hito

Descripción

Antes

Mostrando registros del 1 al 2 de un total de 2 registros

Anterior

1

Siguiente

Continúa en la página siguiente

33

Tabla 3 – Continuación de la página anterior

Detalle Historial Curricular				
Mostrar 10 registros				
Año	Fecha	Hito	Descripción	Antes
2014	28-05-2015	Modificación mayor	Cambio de instituto que dicta la carrera con código CAEV-143	CAEV-143, instituto de ciencias ai
<div><div>C</div><div>Resoluciones</div><div>Código: 9</div><div>Nombre: Comunicación interna 118/2015</div><div>Código: 10</div><div>Nombre: Comunicación interna 161/2015</div><div>Asignaturas</div><div>Código: BOTN143</div><div>Nombre: BOTÁNICA AGRÍCOLA</div></div>				
2015	30-10-2015	Selecione Hito	Agregar requisito en el tercer nivel de la asignatura ICML035	ICML035 no posee requisitos.
Año	Fecha	Hito	Descripción	Antes
Mostrando registros del 1 al 2 de un total de 2 registros				
Anterior				1 Siguiente

Curso normal de eventos	
Acción actor	Respuesta sistema
1.- Este caso de uso comienza cuando un usuario autenticado desea ver el historial curricular de una carrera, para ello debe hacer click en 1.	2.- El sistema despliega la pantalla A, el cual permite al usuario seleccionar facultad, escuela y carrera.
3.- El usuario selecciona la facultad, escuela y carrera, haciendo click en 2,3 y 4.	4.- El sistema crear un texto plano en formato JSON, el cual contiene el detalle solicitado por el usuario.
	5.- El sistema lee el archivo JSON y lo despliega en la tabla B.
6.- Para ver información mas detallada del hito, el usuario puede hacer click en 6.	7.- El sistema despliega la información solicitada en C.
8.- Si el usuario lo desea, puede ver la resolución haciendo click en 7.	9.- El sistema redirecciona al usuario al documento solicitado.

En la Figura 14 se presenta el diagrama de secuencia para este mismo caso de uso, y así mostrar la interacción entre los distintos componentes del software que llevan a cabo esta tarea.

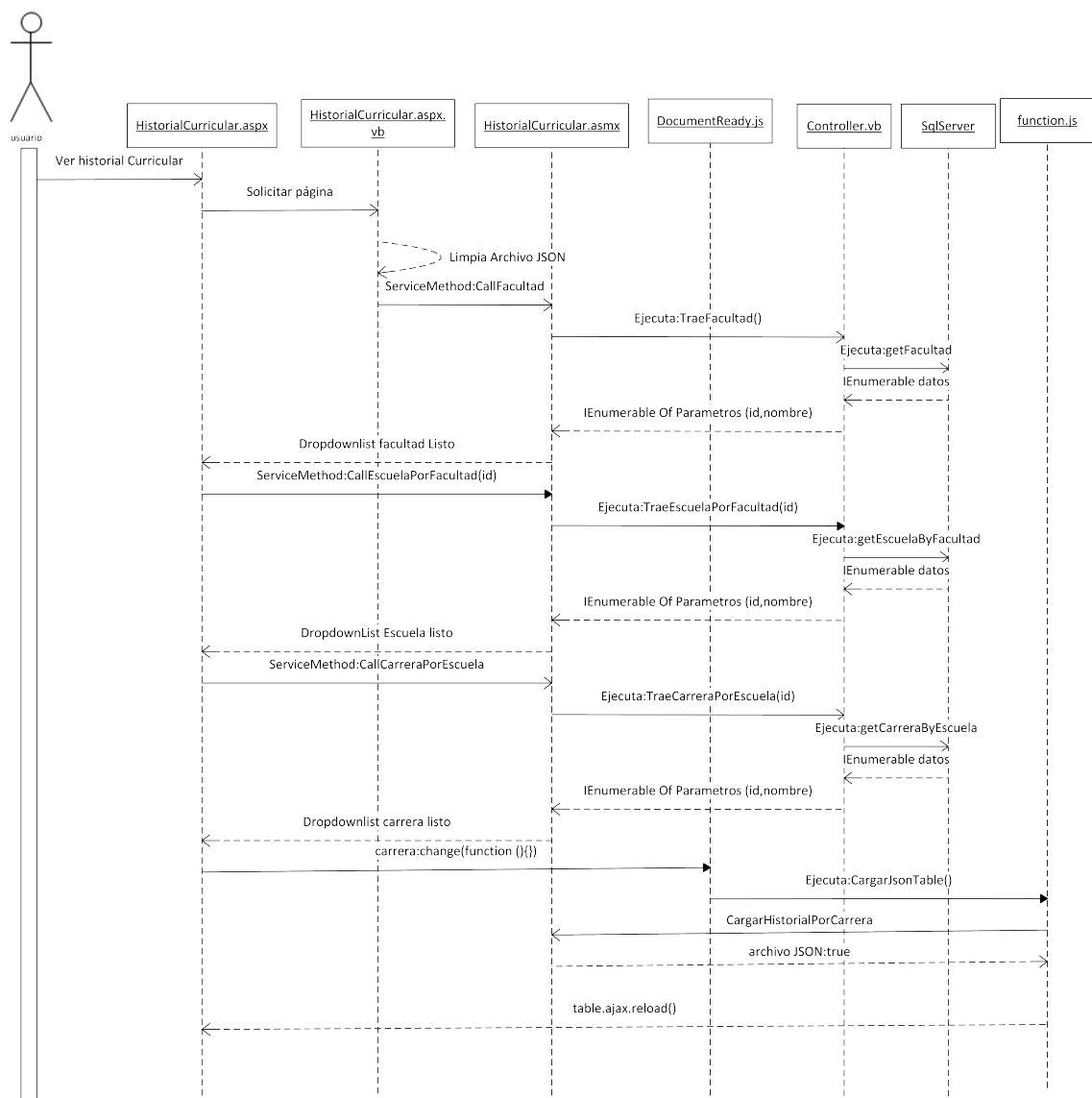


Figura 14. Diagrama de secuencia para el caso de uso *Ver Historial Curricular* ¹⁸

En primer lugar el *HistorialCurricular.aspx* debe asegurarse que el archivo JSON de donde se leen los datos este vacío y con el formato adecuado para su correcta lectura (el formato del archivo JSON utilizado se puede ver en el Anexo A). Una vez realizada esta acción, la vista se comunica con el servicio web (*HistorialCurricular.asmx*) para que la primera lista desplegable, correspondiente a las facultades de la universidad, no este vacía.

El proceso que se describirá a continuación es la secuencia de pasos que realiza el sistema para “poblar” una lista desplegable, por lo que este proceso se repite tres veces, la única diferencia es quién activa el proceso. Para la lista desplegable que se llena con las facultades la activa el sistema antes de mostrar el formulario al usuario, mientras que para las listas desplegables que abarcan a las escuelas y a las carreras las activa el usuario al

¹⁸Elaboración propia.

momento de producir un cambio en las listas desplegables de niveles superiores, es decir, para que el sistema poble la lista desplegable que contiene a las escuelas, el usuario debe producir un cambio en el DropDownList que contiene a las facultades, y para poblar el DropDownList que contiene a las carreras, el usuario debe producir un cambio en la lista desplegable que abarca a las escuelas.

El **poblado de una lista desplegable** comienza cuando el HistorialCurricular.aspx envía un mensaje a la biblioteca Controller.vb indicando qué datos se necesitan de la base de datos (Facultad, Escuela o Carrera), una vez que el Controller.vb recibe la petición, ejecuta la consulta correspondiente y retorna los datos solicitados en formato IEnumerable al servicio web, con el fin de que éste último actor asigne la información retornada en la lista desplegable correspondiente.

Una vez que el sistema haya verificado el estado del archivo JSON y que haya poblado la lista desplegable de las facultades, la plataforma esta en condiciones de mostrar el formulario al usuario para su posterior manipulación. Por otra parte el DocumentReady.js esta a la espera de que el usuario seleccione la carrera, a fin de enviar un mensaje al Function.js con el ID de la carrera.

Una vez que el actor Function.js reciba el ID de la carrera, lo envía al servicio web para que éste cree el archivo JSON, tan pronto como se cree el archivo el HistorialCurricular.aspx envía la variable JSON=true al actor Function.js para que se visualicen los datos del archivo JSON en la interfaz web.

3.3.4. Módulo Gestor de Documentos

El módulo Gestor de documentos permite al administrador y editor administrar el repositorio de documentos de la plataforma, dicho de otra manera, estos usuarios podrán: crear, leer, editar y eliminar registros. Con respecto al usuario de tipo suscriptor, solo poseerá permisos de lectura.

3.3.4.1. Diseño

A continuación se describe el diseño de los dos casos de usos extendidos del módulo de gestión de documentos de la plataforma, que corresponden a: subir documentos y

ver documentos. Se presentará en detalle la composición de estos casos de uso y su participación en el sistema.

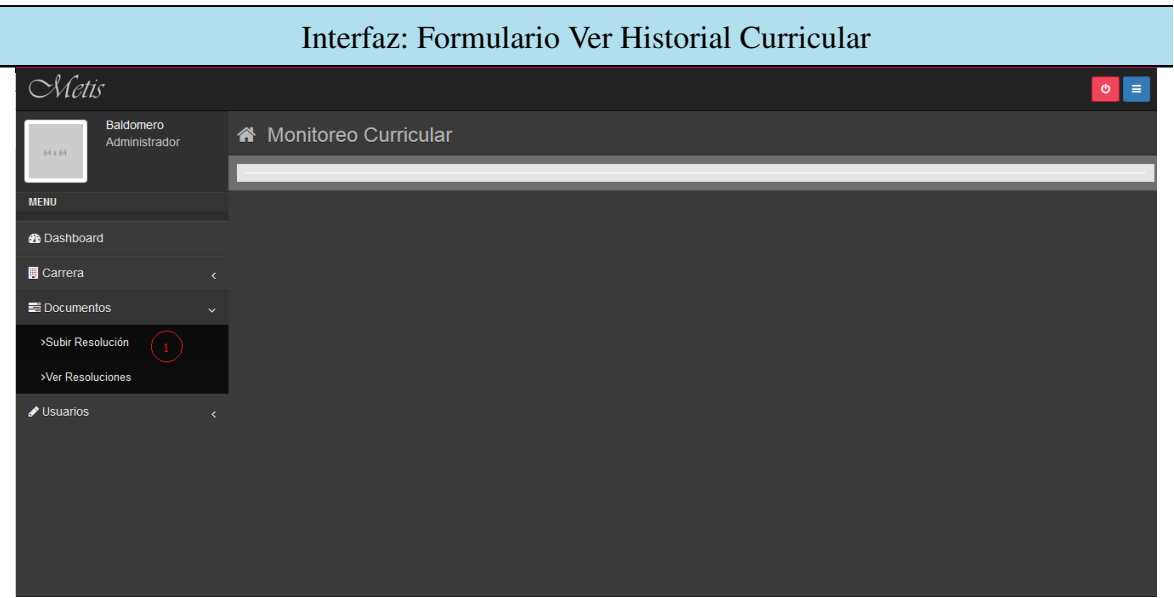
Caso de uso real del módulo gestor de documentos:“Subir documento”

El caso de uso “Subir documento” corresponde al caso de uso extendido de “Gestor de documentos” y ocurre cuando un administrador o un editor desea crear algún hito curricular en un plan de estudios. El caso de uso real “Subir documento” se presenta en la Tabla 4.

Tabla 4. Caso de uso Subir Documento

Caso de uso Subir Documento	
Actores	Administrador y Editor.
Propósito	Crear un hito curricular a un plan de estudios.
Tipo	Primario y esencial

Resumen: El objetivo de este módulo es hacer posible la creación de hitos curriculares y la carga de documentos relacionados con el hito en cuestión al servidor.



Continúa en la página siguiente

Tabla 4 – Continuación de la página anterior

MENU

Dashboard

Carrera

Documentos

Usuarios

Subir Resolución

A

Seleccione Plan

Seleccione Carrera

Fecha de resolución

Hito

Asignaturas

Descripción Cambio

Antes

Después

3

Subir Archivos

Subir Resolución

B

Carga de archivos

Resoluciones

Organizar

Nueva carpeta

acta_consejo_04.pdf

acta_reunion_01.pdf

Comunicacion_interna_003.pdf

Comunicacion_interna_004.pdf

Comunicacion_interna_014.pdf

Comunicacion_interna_022_2.pdf

Comunicacion_interna_022_004.pdf

Comunicacion_interna_028.pdf

Comunicacion_interna_118.pdf

Comunicacion_interna_118_2.pdf

Comunicacion_interna_152-15.pdf

Comunicacion_interna_161.pdf

mail_01.pdf

mail_02.pdf

requerimiento_58484.pdf

requerimiento_58899.pdf

requerimiento_60248.pdf

requerimiento_60365.pdf

solicitud_626.pdf

solicitud_626_2.pdf

Nombre

Adobe Acrobat Document (*.pdf)

Abrir

Cancelar

4

Subir Archivos

Subir Resolución

Seleccione Plan

Seleccione Carrera

Fecha de resolución

Hito

Asignaturas

Descripción Cambio

Antes

Después

5

6

7

Subir Archivos

Continúa en la página siguiente

Tabla 4 – Continuación de la página anterior

Subir Resolución

C

Editar historial

Eliminar historial

Seleccione Plan

2014

Seleccione Carrera

Agronomía

Fecha de resolución

04-11-2015

Hito

Seleccione Hito

Asignaturas

Seleccione las Asignaturas involucradas en el proyecto

Descripción Cambio

Cambio de instituto que dicta la carrera con

Antes

CAEV-143, instituto de ciencias ambientales y evolutiva

Después

PSV143-15 Instituto producción y sanidad vegetal

Resoluciones

Comunicación interna 118/2015

Comunicacion_interna_022_2.pdf

2015 © Baldomero Águla Napoli

Se ha guardado exitosamente el registro!

Curso normal de eventos	
Acción actor	Respuesta sistema
1.- Este caso de uso comienza cuando un usuario de tipo administrador y/o editor desea crear un hito curricular, para ello debe hacer click en 1.	2.- El sistema despliega la pantalla A, el cual permite al usuario introducir información referente al encabezado del hito curricular y subir documentos en formato pdf.
3.- El usuario ingresa toda la información relacionada con el encabezado del hito curricular en 2.	4.- A medida que el usuario va completando los campos, el sistema va validando si están en el formato correcto, es decir, si la información debe ser del tipo: número, sólo letras, longitud máxima, longitud mínima, etc.
5.- Una vez que el usuario haya introducido todos los campos requeridos por el sistema, procede a subir el o los documentos relacionados al hito, haciendo click en 3.	6.- El sistema abre la ventana que se muestra en B.

Continúa en la página siguiente

Tabla 4 – Continuación de la página anterior

7.- El usuario selecciona todos los archivos en formato pdf que desee subir, luego hace click en “Abrir”.	8.- Por cada documento que el usuario desee subir, el sistema agregará a la interfaz gráfica los siguientes elementos; un DropDownList, en el cual el usuario selecciona el tipo de documento qué es; un input, donde el usuario ingresa el número del documento y por último un Label que indica el nombre del documento.
9.- El usuario completa los dos campos nuevos que se agregaron a la interfaz gráfica, para luego hacer clic en “Subir Archivos”.	10.- El sistema valida que los archivos estén en formato correcto.
	11.- El sistema recibe los datos del formulario y los almacena en la base de datos.
	12.- El sistema almacena la operación que se realizó (En este caso creación de un hito) en la bitácora de cambios del sistema.
	13.- El sistema redirecciona al usuario a la pantalla C.
Curso alternativo de los eventos	
	10.- los archivos que el usuario intenta subir no están en formato pdf, o pesan más de 5 mb, por lo que el sistema muestra un mensaje de alerta y el usuario tiene que volver a la etapa Número 5.

En la Figura 15 se presenta el diagrama de secuencia para este mismo caso de uso, y así mostrar la interacción entre los distintos componentes del software que llevan a cabo esta tarea, a continuación se explicará el diagrama presentado.

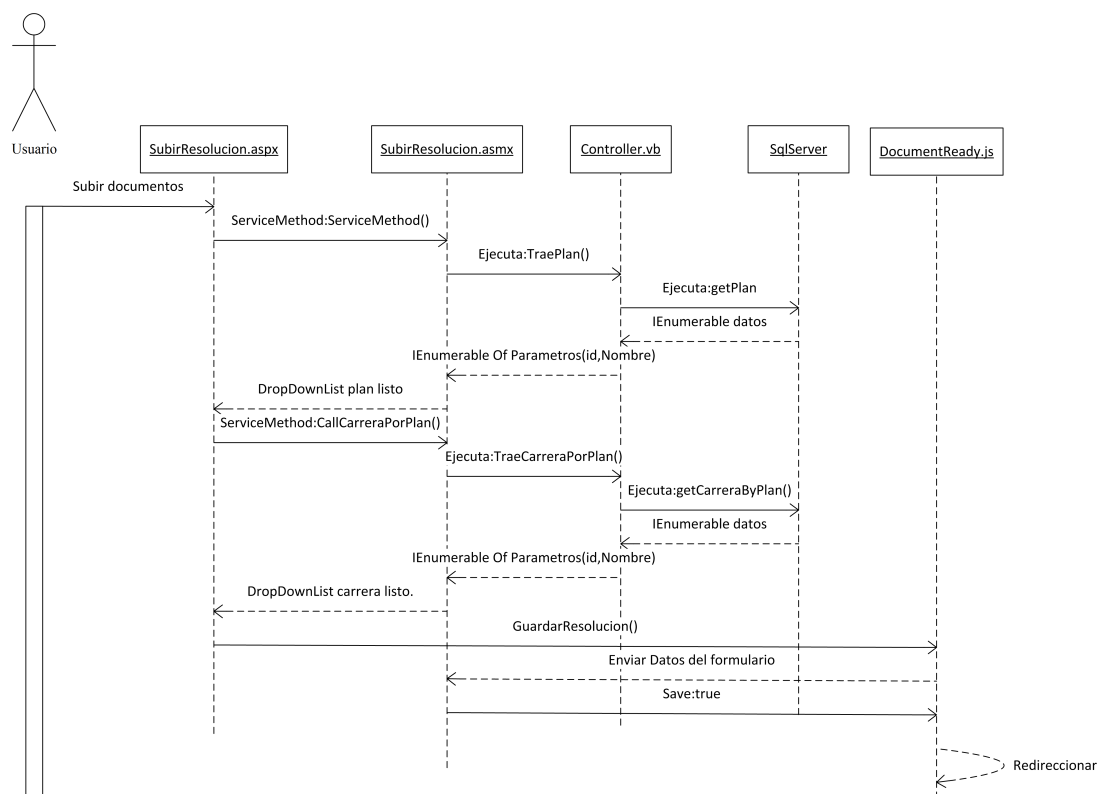


Figura 15. Diagrama de secuencia para el caso de uso *Subir Documentos* ¹⁹

Antes de describir el flujo de control de este caso de uso, hay que mencionar que el módulo Gestor de Documentos trabaja el poblado de los DropDownList del mismo modo que el módulo Historial Curricular, por lo que este proceso no se explicará con mayor detalle en este diagrama.

El flujo de este caso de uso comienza con la carga de la vista SubirResolución.aspx, para ello, antes de mostrar la vista al usuario que realizó la petición, se debe cargar previamente los elementos de las listas desplegables correspondientes a: planes de estudios y asignaturas. Una vez **probada las listas desplegables**(ver Sección 3.3.3.1) el sistema despliega el formulario para que el usuario lo manipule.

Luego de que el usuario hay completado los campos necesarios, y haya seleccionado todos los archivos relacionados con el hito curricular que se desea crear, el usuario envía el mensaje **GuardarResolución** al componente DocumentReady.js, quien es el encargado de enviar los datos del formulario mediante una petición POST al servicio web **SubirResolucion.asmx**.

¹⁹Elaboración propia.

El flujo termina cuando el servicio web envía la variable Sabe:true al DocumentReady.js, con el fin de avisarle a este agente de que los cambios se almacenaron correctamente y que puede redireccionar al usuario a la vista de administración de los hitos curriculares.

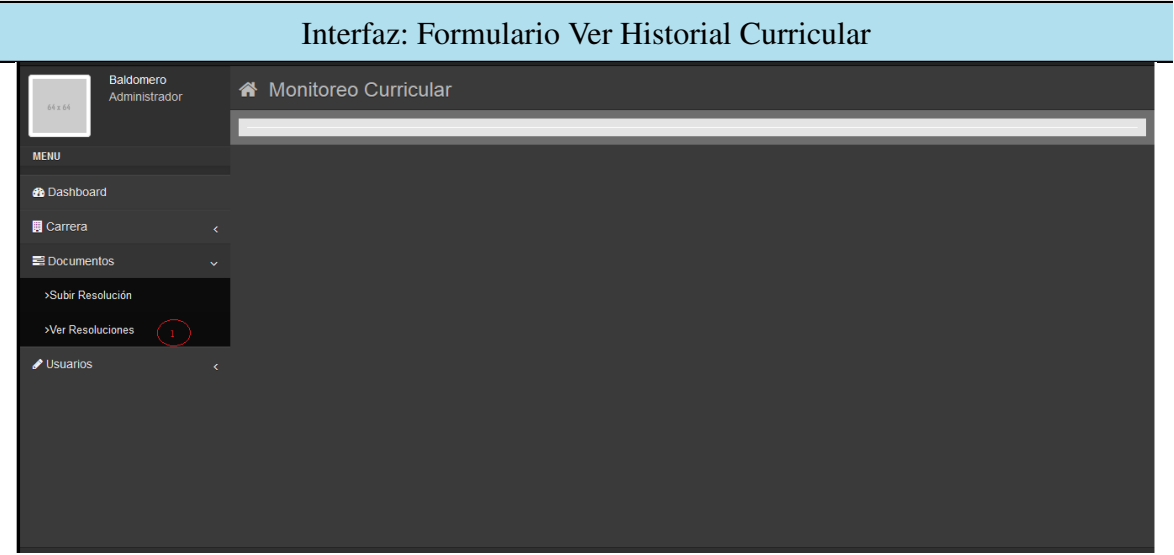
Caso de uso real del módulo gestor de documentos:“Ver documentos”

El caso de uso “Ver documentos” corresponde al segundo caso de uso extendido del caso de uso “Gestor de Documentos”. en este caso de uso todo usuario autenticado tiene acceso y ocurre cuando un algún usuario quiere ver el repositorio de documentos que se encuentra en el sistema. El caso de uso real se presenta en la Tabla 5.

Tabla 5. Caso de uso Ver documentos

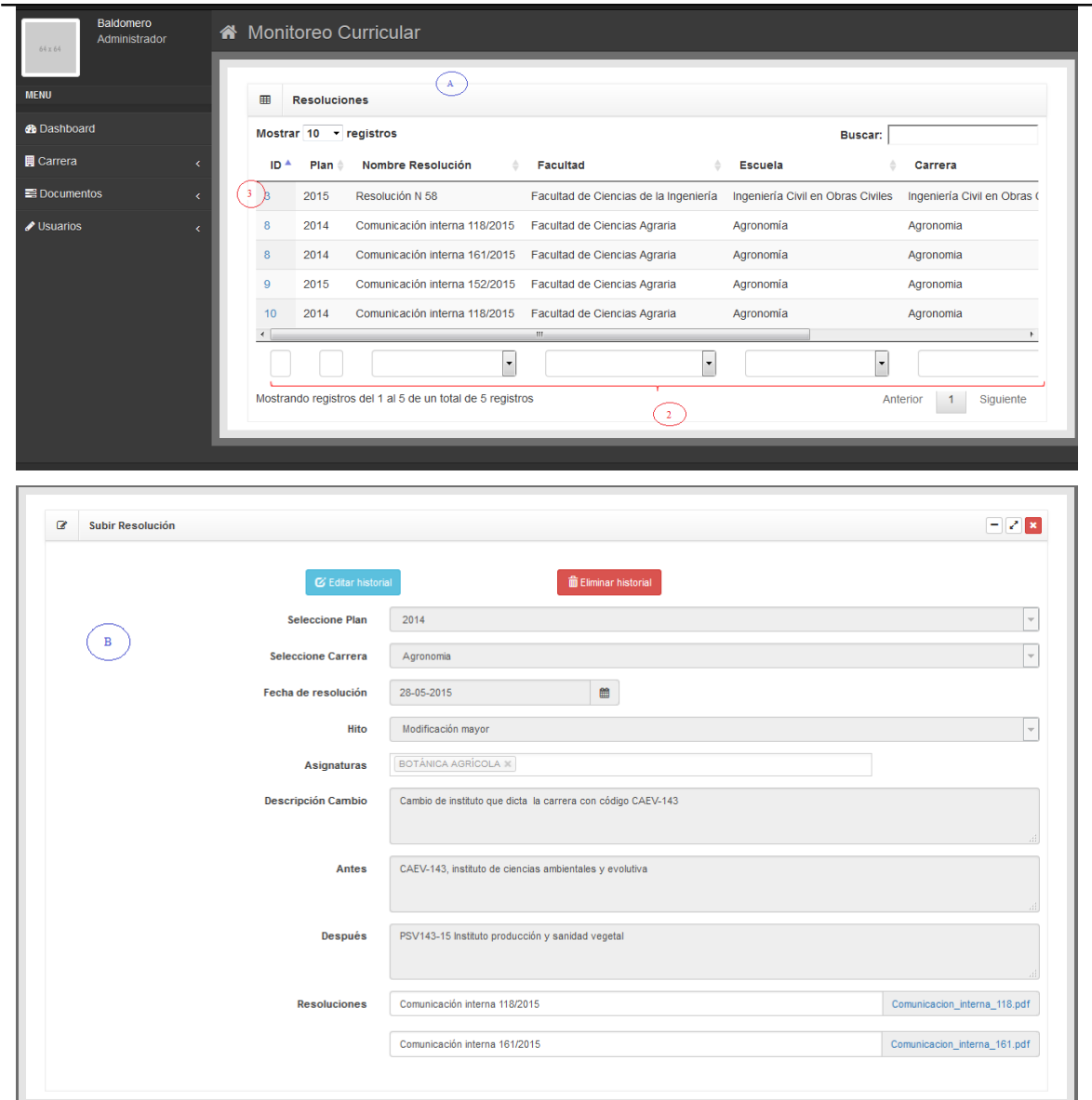
Caso de uso Ver documentos	
Actores	Administrador, Editor y Suscriptor.
Propósito	Visualizar el repositorio de documentos de los cambios curriculares de los planes de estudio.
Tipo	Secundario y esencial

Resumen: El administrador, editor o analista selecciona la sección correspondiente a ver documentos, una vez seleccionado, el sistema mostrará una tabla con todos los documentos de todas las facultades de la universidad. El usuario tendrá opciones de filtros para buscar ya sea por facultad, escuela, carrera y/o tipo de modificación.



Continúa en la página siguiente

Tabla 5 – Continuación de la página anterior



Curso normal de eventos

Acción actor	Respuesta sistema
1.- Este caso de uso comienza cuando un usuario autenticado desea ver el repositorio de documentos, para ello debe hacer click en 1.	2.- El sistema despliega la pantalla A, el cual permite al usuario visualizar una tabla con el registro de los documentos.
3.- El usuario puede hacer algún tipo de filtro avanzado utilizando los DropDownList que se muestran en 2.	4.- El complemento DataTable se encarga de leer los filtros y reducir las filas de la tabla.

Continúa en la página siguiente

Tabla 5 – Continuación de la página anterior

5.- El usuario selecciona el hito que desea visualizar, para ello hace click en el id del registro(3).	6.- El sistema despliega el hito con sus documentos en la ventana que se muestra en B.
---	--

En la Figura 16 se presenta el diagrama de secuencia para este mismo caso de uso, y así mostrar la interacción entre los distintos componentes del software que llevan a cabo esta tarea, a continuación se explicará el diagrama presentado.

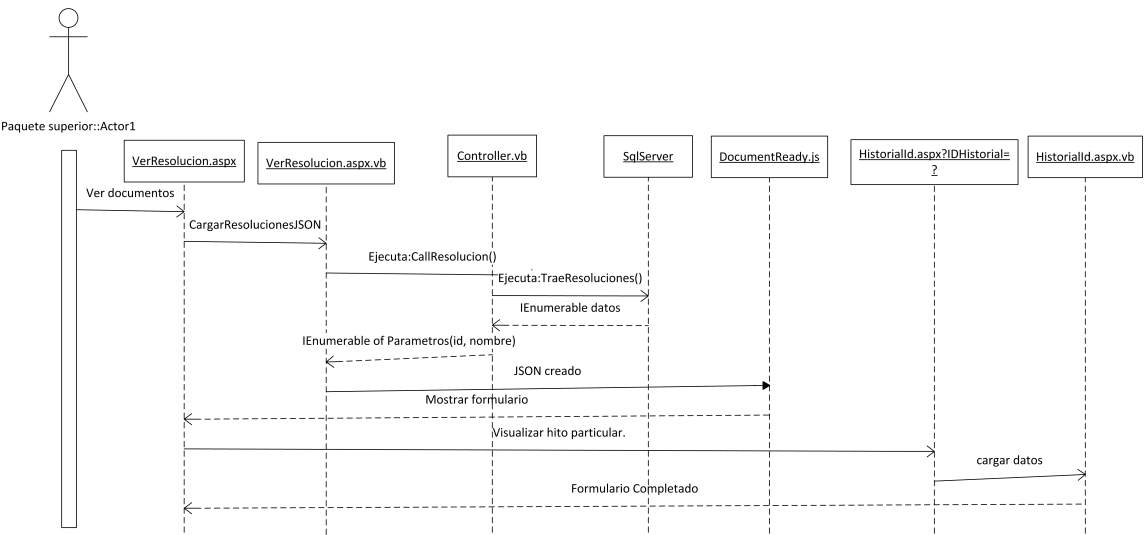


Figura 16. Diagrama de secuencia para el caso de uso *Ver Documentos*²⁰

Cuando un usuario autenticado desea ver algún documento almacenado en la base de datos (resolución, comunicación interna, decretos, etc.), la vista VerResolucion.aspx es la que comienza con toda la secuencia que permite que este caso de uso se ejecute satisfactoriamente.

El primero lugar VerResolucion.aspx solicita los datos de todas las resoluciones a la biblioteca Controller.vb mediante la acción CallResolucion(), una vez que esta biblioteca reciba el mensaje, ejecuta el procedimiento almacenado TraeResoluciones().

Una vez que la consulta se haya efectuado exitosamente, la biblioteca Controller tiene que dar formato a estos datos obtenidos de la base de datos, para que posteriormente se envíen al *code behind*²¹ de la vista VerResolucion.aspx.

²⁰Elaboración propia.
²¹ Archivo que contiene el código ejecutable de las vistas en ASP.NET

Después de que la vista VerResolucion.aspx reciba los datos del Controller.vb, crea el Archivo JSON y se envía una variable JSON = true al agente DocumentReady.js. Al recibir la variable de éxito, DocumentReady.js se encarga de cargar los datos del archivo a la tabla y finalmente mostrar formulario final al usuario autenticado.

4. IMPLEMENTACIÓN

Este capítulo describe las diferentes etapas por las que ha atravesado el desarrollo de este sistema hasta llegar a su última iteración. Se describe una breve reseña de los pasos previos al desarrollo, es decir, la instalación de herramientas y configuración de la máquina de trabajo donde se desarrolló la aplicación web.

4.1. Configuración del ambiente de desarrollo

Como se mencionó en la Sección 3.2.1.2, uno de los requisitos no funcionales es que el proyecto este programado con la estructura con la que trabaja la DTI, dado que el proyecto quedará funcionando en estas dependencias. Para llevar acabo este requisito, el alumno tesista tuvo aproximadamente cinco reuniones con el departamento de desarrollo y mantenimiento de sistemas y de la DTI. El propósito de estas reuniones se describen a continuación.

En primero lugar fue necesario entender el contexto en el cual estaba inmerso el proyecto, por lo que fue necesario contar con todas las entidades de la base de datos de la Universidad que tenían relación directa con el sistema.

Una vez entregado el modelo relacional, se procedió a instalar las herramientas necesarias para el desarrollo en el equipo de trabajo, las herramientas instaladas fueron:

- Microsoft Visual Studio Professional 2013
- Microsoft SQL Server 2008 R2

Durante la instalación de las herramientas mencionadas no hubo mayores problemas.

Finalmente, en conjunto con la Sra. Paola, el alumno tesista creó el “esqueleto” de la plataforma web, el cual estaba separa por las 3 capas de negocio (aplicación web, biblioteca de clases y biblioteca de servicios WCF) y además se realizó la configuración de la conexión a la base de datos.

4.2. Consideraciones Técnicas

Esta sección describe los elementos necesarios para que el sistema funcione de manera óptima:

- alertify v0.3.11
- Metis - Bootstrap-Admin-Template v2.3.2
- daterangepicker v1.3.22
- jQuery bpopup v0.11.0
- DataTables v1.10.7
- jQuery wizard plug-in v3.0.7
- jQuery Input Limiter plugin v1.3.1
- jQuery Rut v0.5
- jQuery Validation Plugin v1.14.0
- jQuery validVal v5.0.2
- Parsleyjs v2.2.0-rc1
- Visual Basic 2013
- Sql Server 2008 R2

4.3. Dificultades de la implementación

Una de las principales dificultades que se presentó al momento de desarrollar la aplicación, fue el trabajar con una arquitectura orientada a servicios, si bien este paradigma esta siendo usado por la mayoría de las aplicaciones web distribuidas, el alumno tesista no tenia experiencia en este tipo de programación, por lo que antes de programar todos los servicios web necesarios para el buen funcionamiento de la aplicación, el alumno tesista tuvo que leer sobre esta paradigma, y esto entorpeció bastante el comienzo del desarrollo.

Otra dificultad detectada mientras se desarrollaba la aplicación, fue entender el paradigma de desarrollo de ASP.NET, en vista de que al usar *code behind*, ASP.NET utiliza sus propios controles, y al momento de compilar, estos controles se transforman en elementos HTML. Así, por ejemplo, el control *DropDownList* de ASP.NET es equivalente a la etiqueta HTML *select*. El trabajar con estos tipos de controles hace que ciertas peticiones (GET, POST, PUT, etc.) se efectúen de forma diferente, y como se mencionó anteriormente, el alumno no poseía ni conocimientos ni experiencia en la la forma en la

que trabaja ASP.NET.

ASP.NET trabaja con la variable *IsPostBack*, el cual nos entrega un valor que indica si la página se está cargando como respuesta a un valor devuelto por el cliente, o si es la primera vez que se obtiene acceso a ella. Esto ocasionó problemas con alguno de los plugins que se incorporaron al proyecto, como por ejemplo el sistema de notificaciones(alertify.js), el cual utiliza una etiqueta *div* como un *popUp* para mostrar un *alert* modificado. Al principio se pensó que el sistema de notificaciones no funcionaba correctamente, ya que los valores de las alertas no se almacenaban, sin embargo después de depurar las acciones sobre las alertas con Firebug, se verificó que el problema no era de plugin, si no que era de la variable *isPostBack*.

En cuanto a la base de datos, el único problema que hubo, fue el tratar de dar un correcto formato a la salida de los procedimientos almacenados. En principio el retorno de los procedimientos era un *1* si el procedimiento se ejecutaba correctamente, o un *-1* si no se ejecutaba, sin embargo este tipo de codificación era bastante ambigua, ya que si salía un *-1*, no se sabía que tipo de error había ocurrido en el sistema, a si que se procedió a trabajar en una codificación mucho más precisa.

Por último, hay que mencionar que una de las tareas que presentó complicaciones, fue la depuración de la aplicación, y esto ocurrió a causa de que las salidas de los procedimientos y servicios web no estaban estandarizados, y al momento de depurar la aplicación no se sabía con exactitud en que sección del código había ocurrido la excepción.

En resumen las mayores dificultades se presentaron en el desconocimiento de las tecnologías utilizadas.

4.4. Presentación de prototipos

El propósito de esta sección, es mostrar el crecimiento del sistema en las diferentes iteraciones. En cada incremento se describen los pasos que efectuaron para finalizar con la iteración exitosamente.

5. VALIDACIÓN DE LA PLATAFORMA

Las pruebas que serán mostradas en este último capítulo, corresponden a una demostración funcional del sistema web. se efectuarán distintas pruebas con el objetivo de verificar el correcto funcionamiento de todos los requisitos funcionales descritos en el Capítulo 3.

5.1. REQF-01: Autenticación de usuario

En la Figura 17 se muestra el inicio de sesión, los campos son: R.U.N del usuario y Contraseña,

Sistema de Apoyo al Monitoreo curricular

Ingrese su R.U.N. y contraseña

probandoLetras

.....

☐ Recordarme

Error, el rut debe ser solo dígitos

Iniciar Sesión

Recuerda que la primera vez que ingreses tu contraseña serán los 6 primeros dígitos de tu RUT

Figura 17. REQF-01: Autenticación de usuario ²²

Se realizaron distintas pruebas para comprobar el correcto funcionamiento de este requisito. A continuación se describen las distintas pruebas realizadas:

- Validación de números: El nick del usuario es su R.U.N. sin el dígito verificador, por lo que el sistema tiene que validar si el usuario ingresa cualquier carácter que no corresponda a un dígito. En caso de que el usuario ingrese algún carácter que no sea dígito, el sistema muestra el siguiente mensaje: **Error, el rut debe ser solo dígitos.**
- Validación nula: Los dos campos solicitados este formulario son campos obligatorios, por lo que si el usuario hace click en iniciar sesión y no ha completado

²²Elaboración propia.

estos campos, el sistema muestra un mensaje de alerta, impidiendo que el servidor realice cálculos innecesarios. Esta validación se ilustra en la Figura 18.

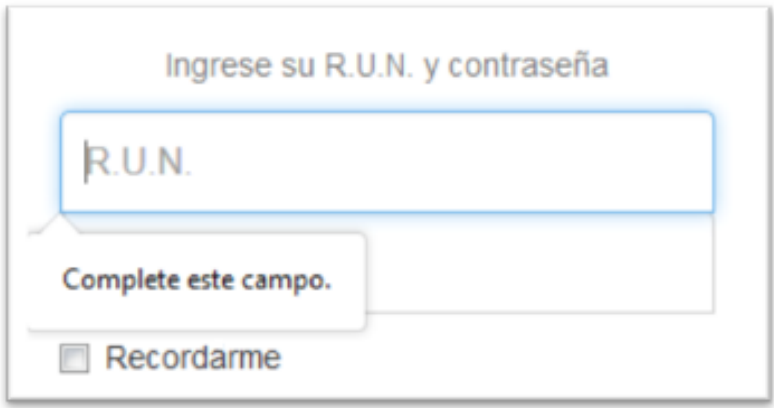


Figura 18. REQF-01: Autenticación de usuario, validación de campos nulos

5.2. REQF-02: Gestión de perfil

Usuarios

Mostrar 10 registros

Buscar:

R.U.N.	Nombre	Apellido paterno	Apellido Materno	Email	rol
4945150	Editor	Editor	Editor	Editor@gmail.com	Editor
17166983	Suscriptor	Suscriptor	Suscriptor	Suscriptor@gmail.com	Suscriptor
17536925	Administrador	Administrador	Administrador	Administrador@gmail.com	Administrador
R.U.N.	Nombre	Apellido paterno	Apellido Materno	Email	rol

Mostrando registros del 1 al 3 de un total de 3 registros

Anterior

1

Siguiente

Figura 19. REQF-02: Gestión de perfil

La Figura 19 ilustra la gestión de perfiles del sistema, este requerimiento solo puede ser accedido por un usuario de tipo administrador. En primer lugar se realizaron pruebas para comprobar que los usuarios del tipo Editor y Suscriptor no tuvieran acceso, para ello se crearon tres usuarios con los tres tipos de perfiles existentes, con el fin de autenticarse y tratar de acceder a este requisito. los resultados muestran en la Tabla 6.

Tabla 6. Resultados de pruebas del acceso a la gestión de perfiles

Tipo de perfil	Resultado de acceso
Administrador	Se pudo acceder exitosamente.
Editor	No se pudo acceder.
Suscriptor	No se pudo acceder.

En segundo lugar se realizaron pruebas para las siguientes operaciones básicas de este requisito: Actualizar usuario y Eliminar usuario. el proceso y los resultados de cada prueba se detallan a continuación.

Actualizar usuario

Uno de los Errores encontrados al momento de validar esta acción, fue que al tratar de actualizar un usuario en particular los datos que se enviaban mediante la petición POST al servidor no eran los correctos. El error detectado se puede apreciar gráficamente en las Figuras 20 y 21.

✎ Crear usuario

R.U.N.

4945150

Nombre

Editor

Apellido paterno

Editor

Apellido materno

Editor

Email

Editor@gmail.com

Rol

Editor

Actualizar historial

Eliminar historial

Figura 20. REQF-02: Gestión de perfil, Actualizar usuarios

Aunque el usuario autenticado esta intentando editar el R.U.N. 4.945.150, en la Figura 21 se puede apreciar que, entre las variables que se están enviando, esta la variable Rut, que corresponde al R.U.N. editado, sin embargo el valor que esta tomando la variable Rut no corresponde al R.U.N. editado, si no que se esta enviando el R.U.N. de la persona autenticada. Este error fue corregido a tiempo.



Figura 21. REQF-02: Gestión de perfi, depuración de los datos enviados mediante la petición POST

Eliminar usuario

Comprobar el correcto funcionamiento del mensaje de alerta que se muestra en la Figura 22 era el principal objetivo de la validación de esta acción, ya que como se mencionó en la Sección 4.3, ASP.NET trabaja de una forma particular las peticiones POST de complementos externos.

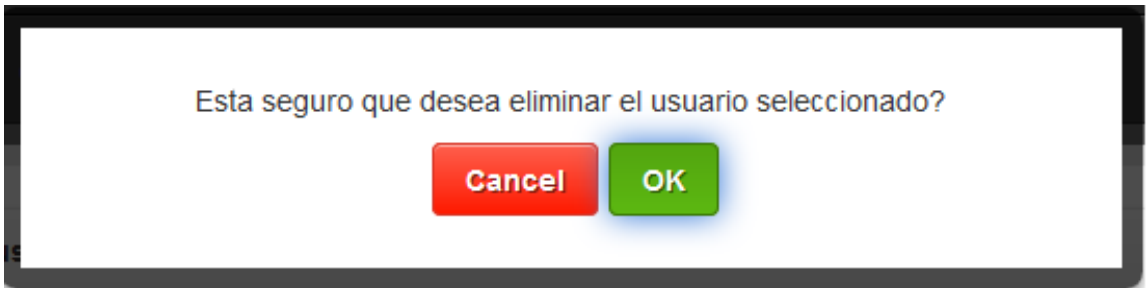


Figura 22. REQF-02: Gestión de perfi, Mensaje de verificación para eliminar algún usuario

La verificación de esta acción se realizó mediante eliminaciones de registro. Se comprobó que los botones de la alerta generada(“Cancel” y “Ok”) realizaran sus respectivas acciones, además, se verificó de que si el registro eliminado poseía algún documento, éste fuera eliminado del servidor.

5.3. REQF-03: Desplegar historial curricular

Este requerimiento no necesitó mayores pruebas puesto que es un requerimiento de visualización.

5.4. REQF-04: Registro de usuarios

Uno de los datos que NO se pueden modificar una vez que se haya creado el usuario, es el R.U.N., por lo que es de vital importancia que este dato se ingrese correctamente al momento de registrar al usuario, para ello se consideró validar la existencia del R.U.N.

ingresado mediante el plugins jQuery.Rut.js.

En caso de que el R.U.N. ingresado no existiera el sistema muestra gráficamente un mensaje de que el R.U.N. no existe (Ver Figura 23), impidiendo que el usuario se almacene en la base de datos.

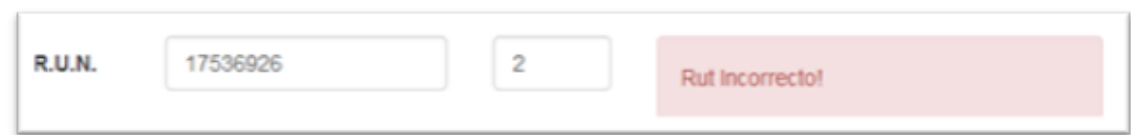


Figura 23. REQF-02: Gestión de perfil

Con respecto a los demás campos se validó que todos los campos sean obligatorios y que el formato del correo electrónico sea un formato válido.

5.5. REQF-05:Gestión de documentos

Se realizaron distintas pruebas para comprobar el correcto funcionamiento de este requisito. A continuación se describen las distintas pruebas realizadas:

- Validación nula: Todos los campos solicitados en este formulario son campos obligatorios, por lo que si el usuario deja un campo sin completar, y luego hace click en Subir archivos, el sistema de forma inmediata marca en rojo todos los campos que no se han completado, y le informa al usuario de que son campos obligatorios.
- Validación de Formato de archivos: Se intentó subir archivos de distintos formatos (.jpg, word, ppt, etc) y el resultado que el sistema muestra un mensaje de Alerta indicando que el formato no es el correcto (Ver Figura 24).
- Validación de eliminación de archivos: Se eliminaron registros curriculares con el fin de verificar que el registro se eliminara en su totalidad del sistema, es decir, que se eliminaran el registro en la base de dato y que se eliminara los archivos vinculados a ese registro eliminado del servidor.

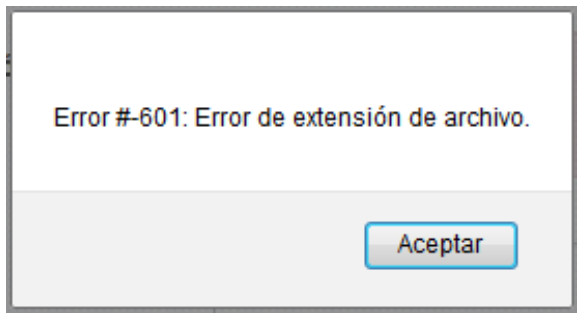


Figura 24. REQF-05: Gestión de documentos, Mensaje de alerta

5.6. REQF-06:Visor de PDF

Este requerimiento no necesitó mayores pruebas puesto que es un requerimiento de visualización.

5.7. REQF-07:Notificaciones

Este requisito se validó en conjunto con los demás requisitos, ya que el sistema de notificaciones esta presente en todo el sistema.

Básicamente el usuario se cercioró de que el sistema de alertas este presente cuando un usuario autenticado realizara alguna operación sobre los datos del sistema.

En la Figura 25 se muestra un ejemplo de una notificación exitosa.

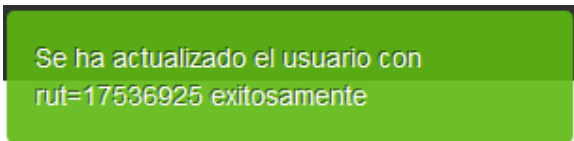


Figura 25. REQF-07: Notificaciones,alerta exitosa

A pesar de que el sistema de notificaciones esta presente en toda la plataforma, éste requisito fue casi uno de los últimos en ser programados, a causa de esto, hubieron muchas acciones que NO tenían su alerta activa.

5.8. REQF-08:Almacenar bitácora de los usuarios

El objetivo de este requisito es facilitar la identificación de la localización de los errores del sistema, por lo que es de vital importancia su correcto funcionamiento.

La validación de este requerimiento es muy distinta a todas las otras validaciones, ya que las validaciones anteriores dependían de los mensajes de alerta que el sistema entregaba, mientras que la validación de este requerimiento depende estrictamente de las salidas de los procedimientos almacenados de la base de datos y de las salidas de los servicios web del sistema, a causa de esto, el alumno tesista se enfocó en validar los procedimientos y los servicios web. Las distintas pruebas realizadas se describirán a continuación.

En primer lugar, el alumno tuvo que validar que las salidas de todos los procedimientos sean estándar, puesto que las salidas se almacenan en la tabla REGISTRO, y esta tabla posee campos obligatorios. El proceso de validación se realizó mediante la ejecución de los procedimientos almacenados y la lectura de las salidas de estos mismos en la consola de SQL Server. Los campos de las salidas de los procedimientos que producen cambios en el sistema son los siguientes:

- **ErrorMessage:** Descripción del mensaje.
- **ErrorMessage:** Descripción del mensaje.
- **ErrorDate:** Fecha en la que ocurrió en evento.

Una vez validados los procedimientos, se procedió a validar los servicios web, para ello fue de gran ayuda el complemento Firebug, ya que el Alumno validó las salidas de los servicios, mediante textos enviados a la consola de Firebug. En la Figura 26, se puede apreciar una salida de un mensaje de error de un servicio web.

Código Error: 547	index.aspx (línea 287)
Mensaje Error: Instrucción INSERT en conflicto con la restricción FOREIGN KEY 'FK_USUARIOS_Rol_6FE99F9F'. El conflicto ha aparecido en la base de datos 'Monitoreo_Curricular', tabla 'dbo.ROLES', column 'ID'	index.aspx (línea 288)
Fecha Error: 2015-11-18 11:32:27.000	index.aspx (línea 289)
Rut: 17536925	index.aspx (línea 290)
Origen Error: Controller/GuardarUsuario	index.aspx (línea 291)

Figura 26. REQF-07: Almacenar bitácora de los usuarios, salida de un procedimiento.

5.9. REQF-09:Visualización de bitácora

Este requerimiento no necesitó mayores pruebas puesto que es un requerimiento de visualización.

6. CONCLUSIONES

La Universidad Austral de Chile se encuentra en una etapa de cambios dado que ya ha empezado con el proceso de innovación curricular. Inicialmente la documentación de estos procesos curriculares se ha almacenado en distintos medios y en varias unidades de la organización. Al comienzo, este modo de gestión funcionó relativamente bien, sin embargo, a medida que el número de carreras innovadas fue creciendo, este tipo de administración fue retrasando la gestión de procesos estratégicos de seguimiento y auto evaluación.

El desarrollo del presente proyecto pretende apoyar la gestión de los procesos curriculares de estudios de pregrado, y como queda en evidencia en el capítulo 3, el objetivo general se ha cumplido en su totalidad.

El sistema no solo permite tener un historial de cambios curriculares en los planes de estudio, si no también permite el almacenamiento centralizado de los documentos relacionados con estos procesos, lo cual permite que la documentación no se deteriore con el tiempo, y también disminuye el tiempo de búsqueda de estos mismos.

Una de las dificultades iniciales identificadas por el autor, fue el desarrollar una aplicación con tecnologías totalmente desconocidas para él. Es verdad que si no hubiera existido esta restricción, no hubiera habido mayores dificultades, sin embargo, el trabajar con tecnologías desconocidas, fue un desafío que se propuso el estudiante para así al finalizar la tesis tener una retroalimentación de qué se siente trabajar bajo presión y sumarle el poco conocimiento sobre las herramientas de desarrollo.

La etapa de “Validación” del proyecto, es muy importante, ya que es inevitable que el sistema inicialmente no falle, puesto que al momento de programar existen muchos factores que hacen que el desempeño de la aplicación no sea el deseado, como por ejemplo el cansancio, el estrés, entre otros factores. Gracias a esta etapa se pudo corregir a tiempo varios errores que se presentaron al momento de probar la aplicación con datos reales.

En síntesis, en el contexto de la educación superior y del cambio de paradigma en el modelo educativo que busca implementar la universidad, esto es, pasar de un modelo

centrado en la enseñanza (profesor) a uno centrado en el aprendizaje (estudiante), es importante que las unidades encargadas de administrar estos procesos posean un sistema informático que facilite no solo la búsqueda de documentos, si no también que almacene la trayectoria de los planes de estudios con el objetivo de tener una referencia de cambios.

6.1. Trabajo futuro

El paso siguiente es la integración total del software desarrollado con los sistemas de la Universidad, para ello se elaboró una propuesta de trabajo, la cual se puede ver en la [Tabla 7](#).

Tabla 7. Propuesta de trabajo

Nombre hito	Descripción	Actores
Reunión	Coordinar una reunión con el encargado de desarrollo de la DTI para crear un ambiente de desarrollo con el fin de que el alumno tenga acceso a los datos reales de la universidad	Alumno tesista, Patrocinante,Encargado de desarrolo de la DTI.
Integración	El alumno tesista tiene que integrar en su totalidad el sistema desarrollado en el ambiente de desarrollo	Alumno tesista.
Reunión	Coordinar una reunión formar con Registro académico, con el fin de que este departamento provea documentación referente a los cambios curriculares	Alumno tesista, Patrocinante, Jefa y secretaria de Registro Académico.
Documentación	Toda la información entregada por registro académico, se debe entregar en formato digital, en caso contrario, se deben escanear todos los documentos	Alumno ayudante.
Poblar BD	Se deben subir todos los documentos facilitados al sistema con el fin de empezar a registrar los historiales curriculares de todas las facultades.	Alumno tesista, Secretaria de Registro Académico.

En cuanto al alcance del proyecto, el prototipo desarrollado podría ser mejorado

incluyendo un módulo estadístico, en el cual se pueda ver de forma gráfica y clara algunas variables, como por ejemplo: las carreras con más cambios curriculares, el porcentaje de carreras innovadas, entre otras variables que se pueden considerar importantes al momento de la generación de informes que apoyen procesos estratégicos de seguimiento.

Finalmente, otra mejora que se podría hacer al prototipo, es aumentar el tipo de archivos soportado, puesto que como la información referente a los cambios curriculares se encuentra en su gran mayoría de forma física, ésta información debe ser escaneada antes de ser subida, y generalmente los software que escanean guardan los documentos en formato imagen y actualmente el software solo permite archivos en formato PDF, por lo que esta mejora supone una disminución de tiempo al momento de querer subir un documento a la plataforma.

7. REFERENCIAS

Referencias

- [Ale15] AlertifyJS. Disponible en <http://alertifyjs.com/>. Consultado el 04 de Noviembre de 2015.
- [Alt7] Altamirano P. & Beltrán C. et al (2007). *Modelo educacional y enfoque curricular*. Valdivia, Chile
- [Boo15] Bootstrap. Disponible en <http://getbootstrap.com/about//>. Consultado el 04 de Noviembre de 2015.
- [Dep15] Departamento de Admisión y Matrícula. Disponible en <http://www.uach.cl/organizacion/direccion-de-pregrado/admision-y-matricula/departamento-de-admision-y-matricula>. Consultado el 05 de Mayo de 2015.
- [Dac15] Departamento de Aseguramiento de la Calidad e Innovación Curricular (DACIC). Disponible en <http://www.uach.cl/organizacion/direccion-de-pregrado/dacic>. Consultado el 07 de Mayo de 2015.
- [Dir15] Dirección de estudios de pregrado. Disponible en <http://www.uach.cl/organizacion/direccion-de-pregrado/registro-academico>. Consultado el 05 de Mayo de 2015.
- [Eje15] EjemplosTIW. Patrón de arquitectura Modelo Vista Controlador (MVC). Disponible en <http://www.lab.inf.uc3m.es/~a0080802/RAI/mvc.html>. Consultado el 09 de Noviembre de 2015.
- [Gar13] García D (2013). WCF (I): Introducción a Windows Communication Foundation Disponible en <https://danielggarcia.wordpress.com/2013/12/11/wcf-i-introduccion-a-windows-communication-foundation/>. Consultado el 15 de Diciembre de 2015.
- [GitH15] Git. Acerca del control de versiones Disponible en <https://git-scm.com/book/es/v1/Empezando-Acerca-del-control-de-versiones>. Consultado el 18 de Noviembre de 2015.
- [Git15] GitHub. github/onokumus Disponible en <https://github.com/onokumus/Bootstrap-Admin-Template#toc>. Consultado el 04 de Noviembre de 2015.
- [Jof15] Jofré M (2013). *Qué es un template*. Disponible en <http://www.glidea.com.ar/blog/que-es-un-template>. Consultado el 04 de Noviembre de 2015.
- [JQu15] JQuery Disponible en <https://jquery.com/>. Consultado el 03 de Noviembre de 2015.
- [Mic15a] Microsoft. Información general sobre ASP.NET Disponible en <https://msdn.microsoft.com/es-es/library/dd566231.aspx>. Consultado el 10 de Noviembre de 2015.
- [Mic15b] Microsoft. Usar SQL Server Management Studio Disponible en <https://msdn.microsoft.com/es-es/library/ms174173%28v=sql.120%29.aspx>. Consultado el 10 de Noviembre de 2015.

- [Mic15c] Microsoft SQL Server 2008 Express Disponible en <https://www.microsoft.com>. Consultado el 05 de Noviembre de 2015.
- [Moz15] Mozilla. Complementos Disponible en <https://addons.mozilla.org/es/firefox/addon/firebug/>. Consultado el 18 de Noviembre de 2015.
- [Par15] Parsleyjs. Disponible en <http://parsleyjs.org/doc/about.html>. Consultado el 05 de Noviembre de 2015.
- [Tin10] Tinoco O. & Rosales P. & Salas J (2010). *Criterios de selección de metodologías de desarrollo de software*. Disponible en <http://www.redalyc.org/articulo.oa?id=81619984009>. Consultado el 12 de Noviembre de 2015.
- [Tum11] Tumanoff R. & Chauriye S (2011). *Innovación curricular en las universidades del consejo de rectores 2000-2010*. Santiago, Chile.

8. ANEXOS

Anexo A: Formato inicial de los archivos JSON.

```
1 {  
2   "data": []  
3 }  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24
```

Figura 27. Formato inicial de los archivos JSON.

Anexo B: CascadingDropDown

CascadingDropDown es una extensión de ASP.NET AJAX que se puede conectar a un DropDownList para obtener la población automática de un conjunto de controles DropDownList. Cada vez que se selecciona uno de los DropDownList, el CascadingDropDown realiza una llamada a un servicio web, para recuperar la lista de valores para poblar el siguiente DropDownList.

Propiedades:

- TargetControlID: El ID de la lista desplegable a la que se aplicará.
- Category: se define como el nombre de la categoría que la lista desplegable representa. Su utilidad será la de representar uno de los dos parámetros de entrada al ServiceMethod que estudiaremos posteriormente.
- PromptText: Es un texto opcional que verá el usuario cuando la lista desplegable esté vacía.
- LoadingText: Es un texto opcional que verá el usuario cuando el dato se está cargando.
- ServicePath: Es el path del servicio web que devuelve la información que se usará para rellenar la lista desplegable.

- ServiceMethod: Es el nombre del método el cual pobla la lista desplegable.
- ParentControlID: ID de la lista desplegable de cuya selección depende esta lista desplegable.
- SelectedValue: valor que vendría seleccionado por defecto. Es opcional.

La función a la que se llamará para rellenar la lista desplegable tendrá el siguiente formato:

```
[ WebMethod ]

public CascadingDropDownNameValue[] GetDropDownContents( string
    knownCategoryValues , string category ){

    ...
}
```

Se puede observar que:

- La función debe ir precedida por [WebMethod].
- CascadingDropDownNameValue es un tipo de dato dentro del namespace AjaxControlToolkit.
- El segundo parámetro (category) corresponde al atributo Category que se ha definido previamente en el control CascadingDropDown.

Anexo C: Diccionario de datos.

Nombre Tabla	ROLES		
Descripción	Almacena los distintos tipos de roles del sistema		
Nombre campo	Descripción	Tipo	Llave
ID	Identificador único de la tabla	INT	Primary Key
Nombre	Nombre del rol	VARCHAR	

Figura 28. Tabla ROLES

Nombre Tabla	USUARIOS		
Descripción	Almacena los usuarios del sistema		
Nombre campo	Descripción	Tipo	Llave
RUT	Rut del usuario	INT	Primary Key
Nombre	Almacena el nombre del usuario	VARCHAR	
ApPaterno	Almacena el apellido paterno del usuario	VARCHAR	
ApMaterno	Almacena el apellido materno del usuario	VARCHAR	
Password	Almacena la contraseña del usuario	VARCHAR	
Email	Almacena el correo electrónico del usuario	VARCHAR	
CreatedDate	Almacena la fecha de creación del usuario	DATETIME	
LastLoginDate	Almacena la última fecha de autenticación	DATETIME	
Rol	Almacena el tipo de rol que el usuario posee	INT	Foreign Key de la tabla ROLES

Figura 29. Tabla USUARIOS

Nombre Tabla	REGISTRO		
Descripción	Almacena todos los eventos que el usuario realiza en el sistema(Crear, Eliminar, Actualizar)		
Nombre campo	Descripción	Tipo	Llave
IDREGISTRO	Identificador único de la tabla REGISTRO	INT	Primary Key
CodigoError	Almacena el código del evento	VARCHAR	
MensajeError	Almacena la descripción del evento	VARCHAR	
FechaError	Almacena la fecha en la que ocurrió el evento	DATETIME	
Rut	Almacena el rut del usuario que ejecutó el evento	INT	Foreign Key de la tabla USUARIOS
OrigenError	Almacena el path del sitio web en donde ocurrió el evento	VARCHAR	
IdObjetos	Almacena el identificador único de registro involucrado en el evento	int	

Figura 30. Tabla REGISTRO

Nombre Tabla	HISTORIAL_CURRICULAR		
Descripción	Almacena todos los hitos curriculares que se efectúan en una carrera		
Nombre campo	Descripción	Tipo	Llave
IDHISTORIALCURRICULAR	Identificador único de la tabla HISTORIAL_CURRICULAR	smallint	Primary Key
IDPLAN	Identificador del plan de estudio	INT	Foreign Key de la tabla PLANESTUDIO
IDCARRERA	Identificador del plan de la carrera	INT	Foreign Key de la tabla CARRERAS
FECHARESOLUCION	Almacena la fecha en la que se sube el documento	DATETIME	
HITO	Almacena el tipo de modificación que se está efectuando	VARCHAR	
DESCRIPCIONCAMBIO	Almacena una breve descripción del cambio efectuado	VARCHAR	
ANTES	almacena los datos del plan de estudio antes del cambio	VARCHAR	
DESPUES	Almacena los datos del plan de estudios después del cambio	VARCHAR	

Figura 31. Tabla HISTORIAL CURRICULAR

Nombre Tabla	RESOLUCIONES		
Descripción	Almacena todos los documentos relacionados con los cambios curriculares		
Nombre campo	Descripción	Tipo	Llave
ID_RESOLUCION	Identificador único de la tabla RESOLUCIONES	smallint	Primary Key
IDHISTORIALCURRICULAR	Identificador del historial curricular	smallint	Foreign Key de la tabla HISTORIAL_CURRICULAR
NOMBRERESOLUCION	Almacena el nombre de la resolución	VARCHAR	
PATH_RESOLUCION	Almacena la ubicación física de la resolución	VARCHAR	

Figura 32. Tabla RESOLUCIONES

Nombre Tabla	ASIGNATURAS_HISTORIAL		
Descripción	Almacena LAS ASIGNATURAS RELACIONADAS CON LOS HISTORIALES CURRICULARES		
Nombre campo	Descripción	Tipo	Llave
IDHISTORIALCURRICULAR	Identificador del historial curricular	smallint	Foreign Key de la tabla HISTORIAL_CURRICULAR
IDASIGNATURA	Identificador de la asignatura	smallint	Foreign Key de la tabla ASIGNATURAS

Figura 33. Tabla ASIGNATURA HISTORIAL