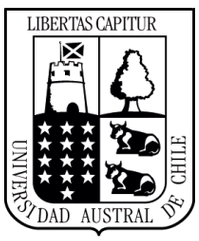


Universidad Austral de Chile

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
—

PROFESOR CO-PATROCINANTE:
????
INGENIERO CIVIL EN INFORMÁTICA
MAGÍSTER EN INFORMÁTICA EDUCATIVA

PROFESOR INFORMANTE:
N.N.
N.N.

BALDOMERO ÁGUILA NAPOLI

VALDIVIA - CHILE
2015

AGRADECIMIENTOS

ÍNDICE

ÍNDICE DE TABLAS

TABLA PÁGINA

ÍNDICE DE FIGURAS

FIGURA PÁGINA

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"[MOD07]. 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 mas importantes en este ámbito, es la gestión de los proyectos curriculares de la carrera. 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 en cual están constantemente manipulando información curricular de las carreras, estos departamento 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.

Las ventajas de contar con una plataforma web que almacene datos históricos de las carreras, es disminuir el trabajo que poseen estos departamentos al momento de requerir

alguna información curricular.

ABSTRACT

1. INTRODUCCIÓN

"Durante la última década, las universidades pertenecientes al Consejo de Rectores de Universidades chilenas (CRUCH) han promovido diversas iniciativas de Innovación Curricular"[INN11]. 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 del Dr. Mauricio Ruiz-Tagle quien observó las falencias que tienen los departamentos ya indicados 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 anteriormente 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 las y 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]

[1]Elaboración propia.

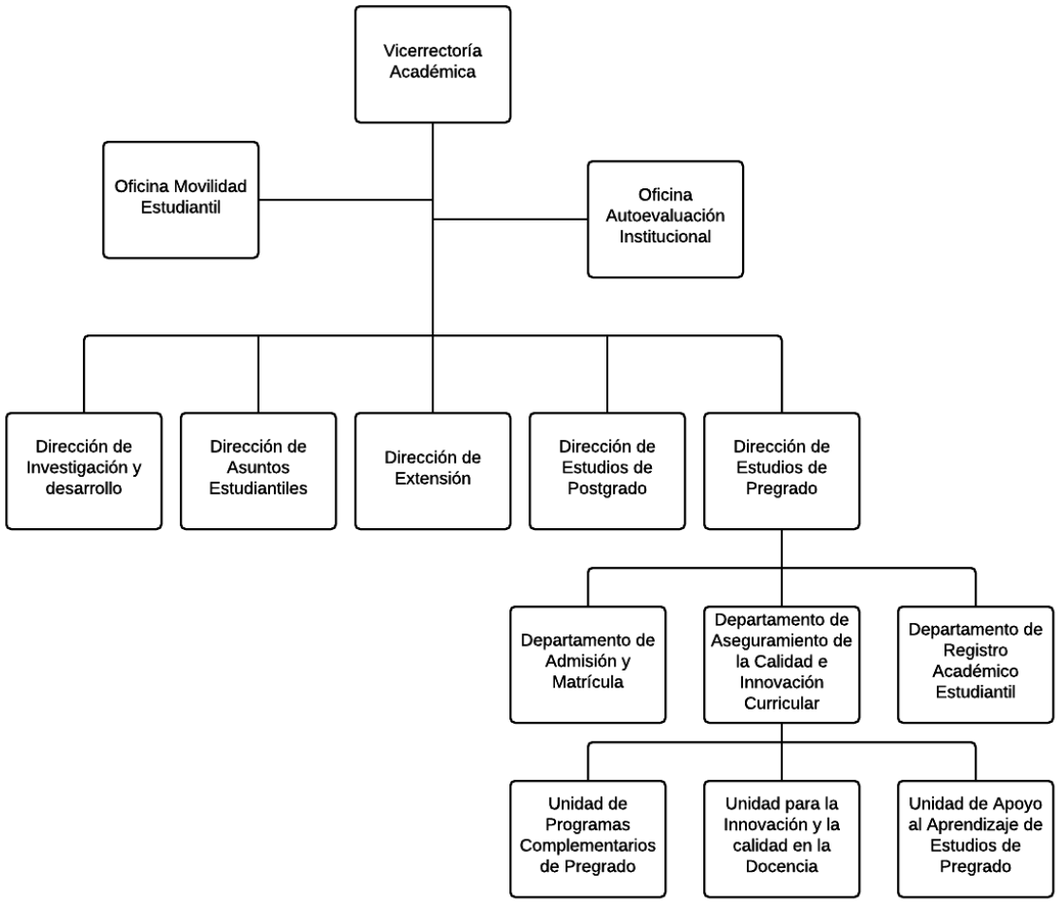


Figura 1. Organigrama Vicerrectoría y DACIC [1]

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

Como su nombre lo indica, este departamento esta presente cuando el alumno ingresa a la Universidad Austral de Chile, en cualquiera de las siguientes modalidades:

- 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, esta a cargo de la Sra. 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 3 describe la secuencia básica que se lleva a cabo para la creación de nuevas carreras en la Universidad Austral de Chile.

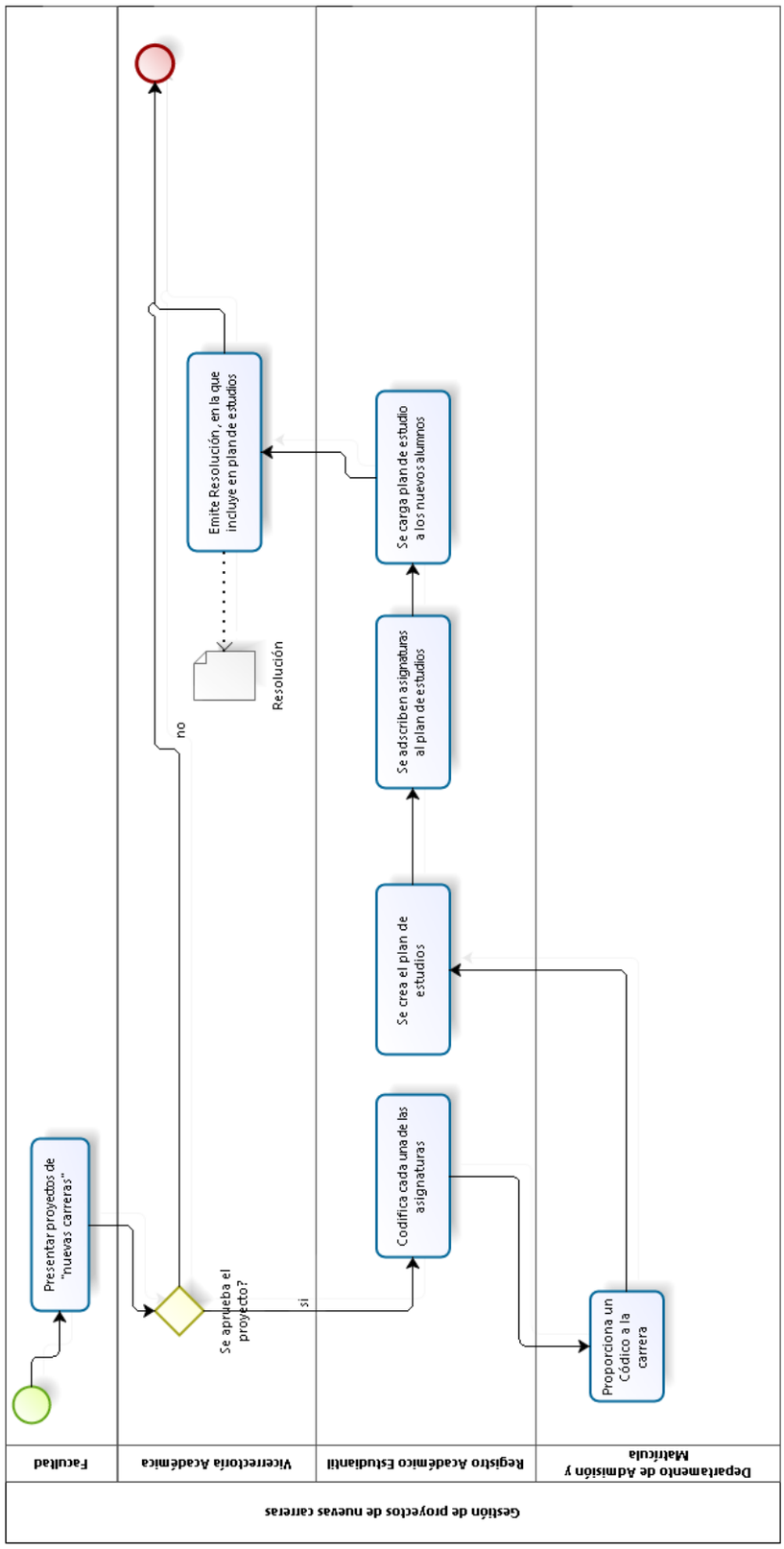


Figura 2. Proceso de presentación de nuevos proyectos [2]

Las fechas para la presentación de nuevos proyectos o innovaciones curriculares, se

[2]Elaboración propia.

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.

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 especifica. 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)
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 estudio, 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 estudio 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 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 ultimo Registro Académico Estudiantil realizan una petición de requerimientos al 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 Pregreado rechazó o aprobó las modificaciones curriculares.

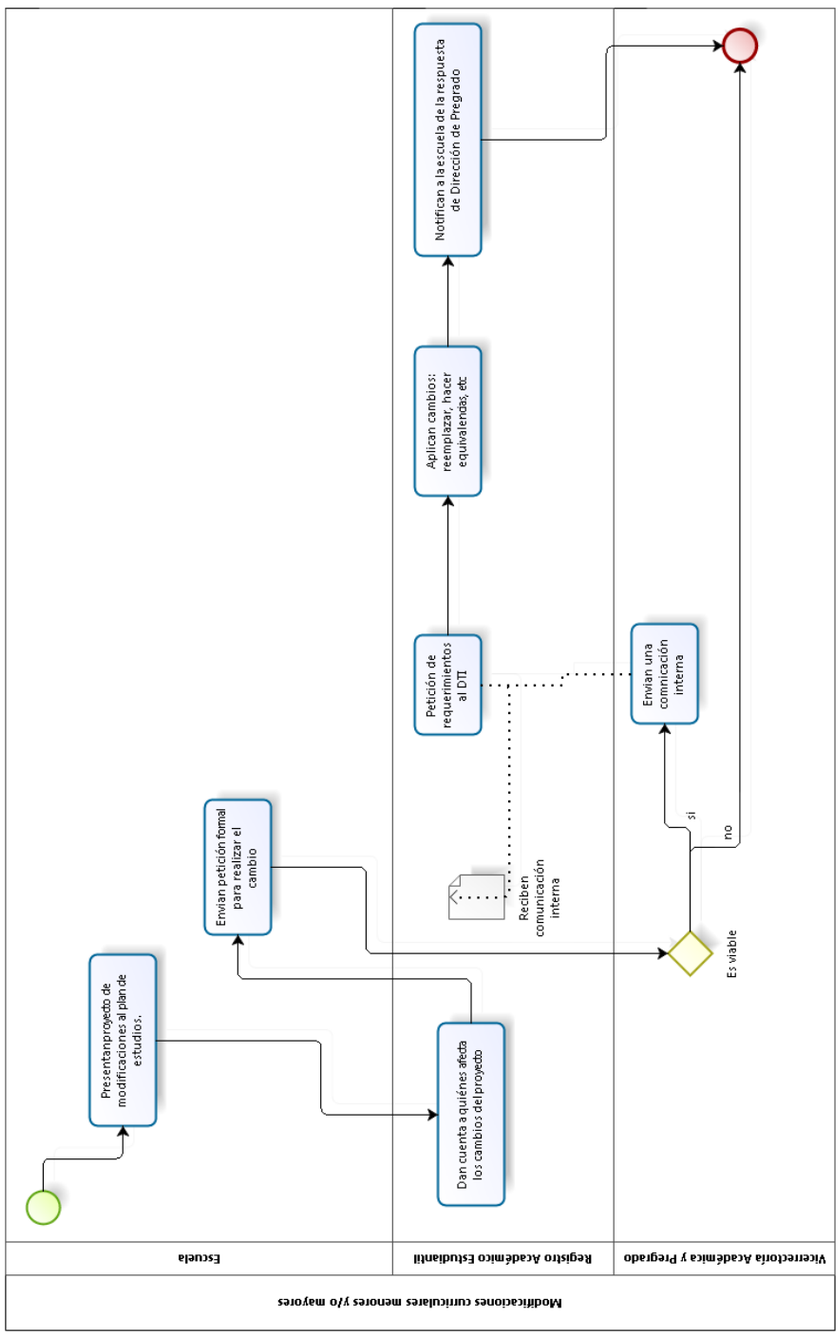


Figura 3. Procesos de cambios Curriculares Mayores y/o menores [3]

[3]Elaboración 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 innovadas, etc.

Para finalizar la etapa teórica, se investigó sobre las tecnologías en las cual se desenvolverá 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 dos incrementos. En el primer incremento se seleccionaron los requisitos que se relacionaban con la gestión del historial curricular de una carrera en particular, mientras que en el segundo incremento se seleccionan los requisitos referentes a la gestión de documentos.

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 utilizo el software y ayudo 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 DTI[4] 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 MVC[5] 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.

[4]Dirección de tecnologías e información
[5]Modelo-vista-Controlador

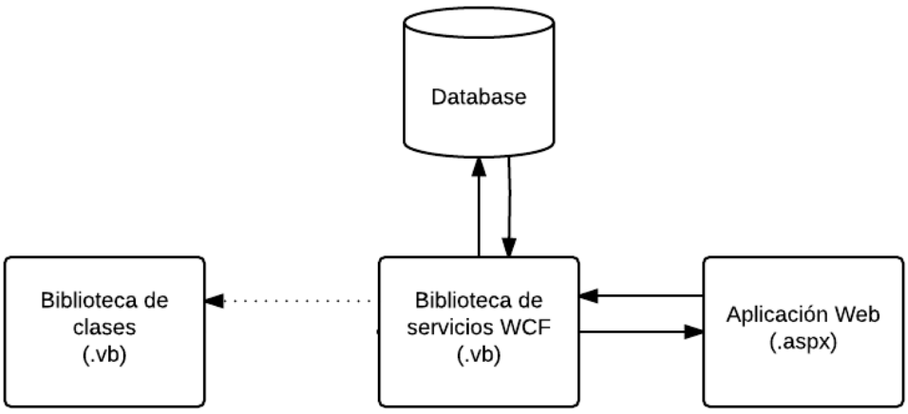


Figura 4. Arquitectura pseudo MVC Universidad Austral de Chile [6]

En la Figura 4 se puede apreciar las 3 capas de esta arquitectua. 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 WFC, 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.

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.

[6]Elaboración propia.

2.2. Tecnologías para el desarrollo de la plataforma

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.

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 [7] 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

[7] Ajax: Asynchronous JavaScript And XML

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 [8]

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 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 [gli15].

[8] **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.

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 [mic15].

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"[mic15].

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# [inf15].

ASP.NET introduce el concepto del code-behind en el desarollo 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 de 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 [GIT15], 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

2.3.1. Resolución

2.3.2. Comunicación interna

2.3.3. Especificación de requerimientos

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 [GOM10]. 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.

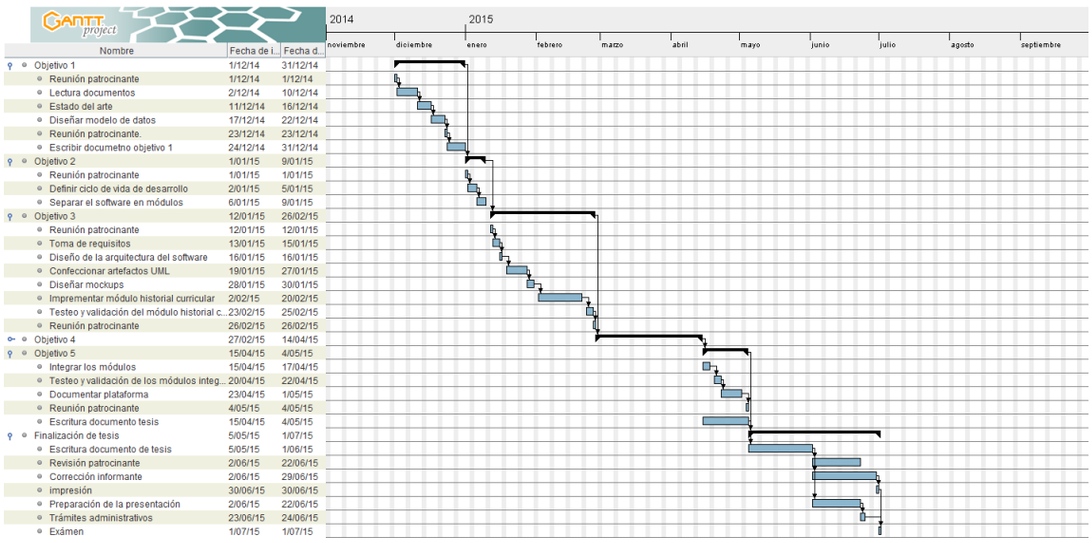
En pocas palabras las metodologías ayudan a tener una buena administración y gestión sistemática de todo proyecto de software, y llevarlo a cabo con altas posibilidades de éxito. De esta forma es posible crear, desarrollar y mantener un sistema desde que surge la necesidad del producto hasta que se cumple 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 5 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.



Nombre	Fecha de i...	Fecha d...
Objetivo 1	1/12/14	31/12/14
Reunión patrocinante	1/12/14	1/12/14
Lectura documentos	2/12/14	10/12/14
Estado del arte	11/12/14	16/12/14
Diseñar modelo de datos	17/12/14	22/12/14
Reunión patrocinante.	23/12/14	23/12/14
Escribir documetno objetivo 1	24/12/14	31/12/14
Objetivo 2	1/01/15	9/01/15
Reunión patrocinante	1/01/15	1/01/15
Definir ciclo de vida de desarrollo	2/01/15	5/01/15
Separar el software en módulos	6/01/15	9/01/15
Objetivo 3	12/01/15	26/02/15
Reunión patrocinante	12/01/15	12/01/15
Toma de requisitos	13/01/15	15/01/15
Diseño de la arquitectura del software	16/01/15	16/01/15
Confeccionar artefactos UML	19/01/15	27/01/15
Diseñar mockups	28/01/15	30/01/15
Impremental módulo historial curricular	2/02/15	20/02/15
Testeo y validación del módulo historial c...	23/02/15	25/02/15
Reunión patrocinante	26/02/15	26/02/15
Objetivo 4	27/02/15	14/04/15
Objetivo 5	15/04/15	4/05/15
Integrar los módulos	15/04/15	17/04/15
Testeo y validación de los módulos integ...	20/04/15	22/04/15
Documentar plataforma	23/04/15	1/05/15
Reunión patrocinante	4/05/15	4/05/15
Escritura documento tesis	15/04/15	4/05/15
Finalización de tesis	5/05/15	1/07/15
Escritura documento de tesis	5/05/15	1/06/15
Revisión patrocinante	2/06/15	22/06/15
Corrección informante	2/06/15	29/06/15
impresión	30/06/15	30/06/15
Preparación de la presentación	2/06/15	22/06/15
Trámites administrativos	23/06/15	24/06/15
Exámen	1/07/15	1/07/15

Figura 5. Carta Gantt que exhibe las etapas de desarrollo del sistema [9]

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.

[9] Elaboración 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	Autentificación 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.
REQF-04	Registro de usuarios

Continúa en la página siguiente

Tabla 1 – *Continuación de la pagina anterior*

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-07	Visor de PDF
Descripción	El sistema debe permitir al usuario ver cualquier documento que se encuentre almacenado en la base de datos.
REQF-08	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-09	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-10	Visualización de bitácora
Descripción	El sistema debe permitir al administrador visualizar todos los eventos ocurridos en el sistama.

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.

3.2.2. Modelo conceptual

Para facilitar la compresión de la problemática que se desea solucionar mediante el producto software, en la Figura 6 se presenta un modelo conceptual, que muestra el dominio dentro del cual se encuentra inmerso el proyecto.

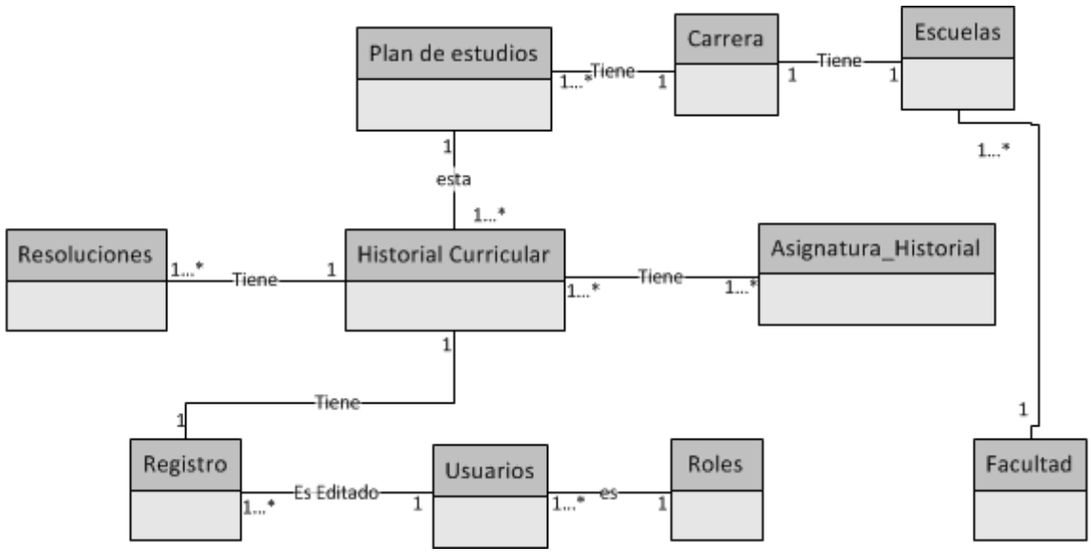


Figura 6. Modelo conceptual del proyecto [10]

3.2.3. Casos de uso

En las Figuras 7, 8, 9 se presentan los diferentes casos de uso para los actores involucrados en el sistema.

[10]Elaboración propia.

Figura 7. Diagrama de caso de uso para el Administrador [11]

[11] Elaboración propia.

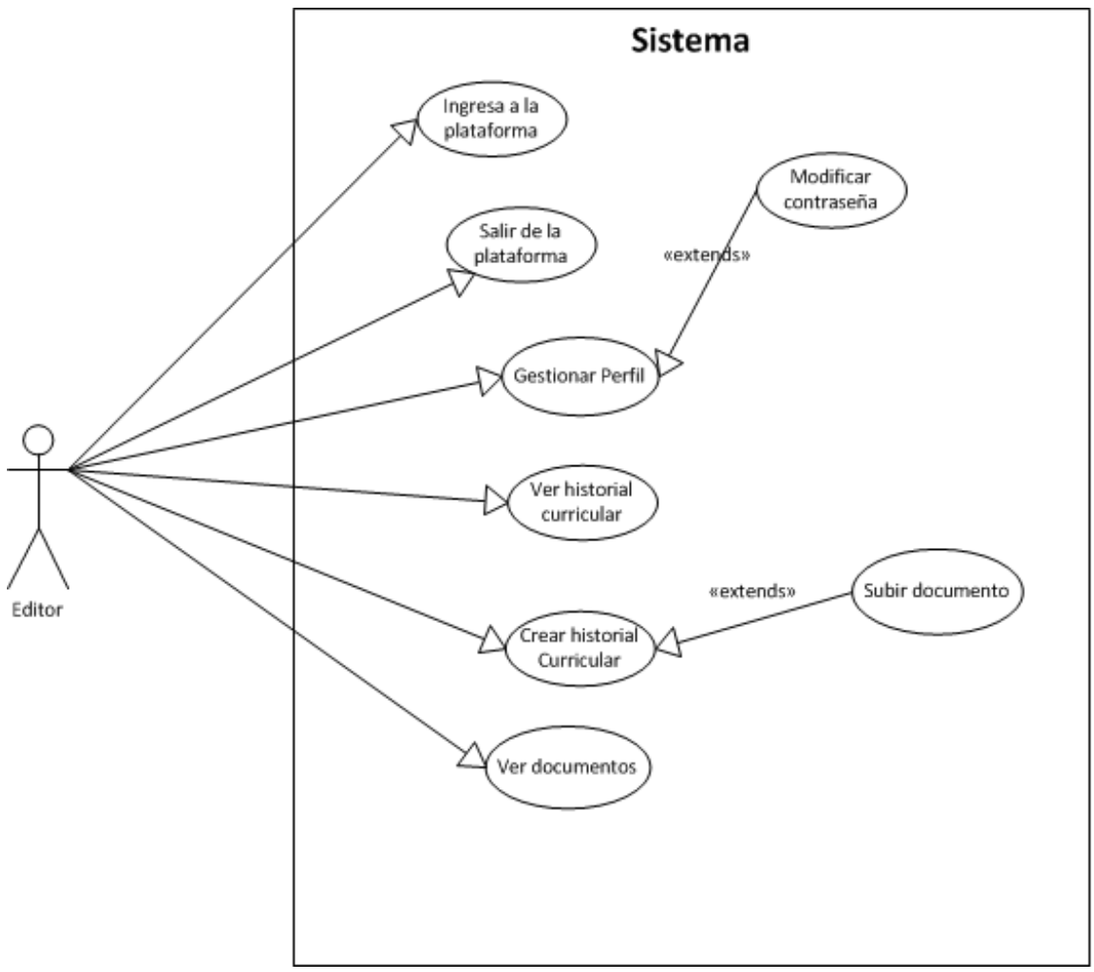


Figura 8. Diagrama de caso de uso para el Editor [12]

[12]Elaboración propia.

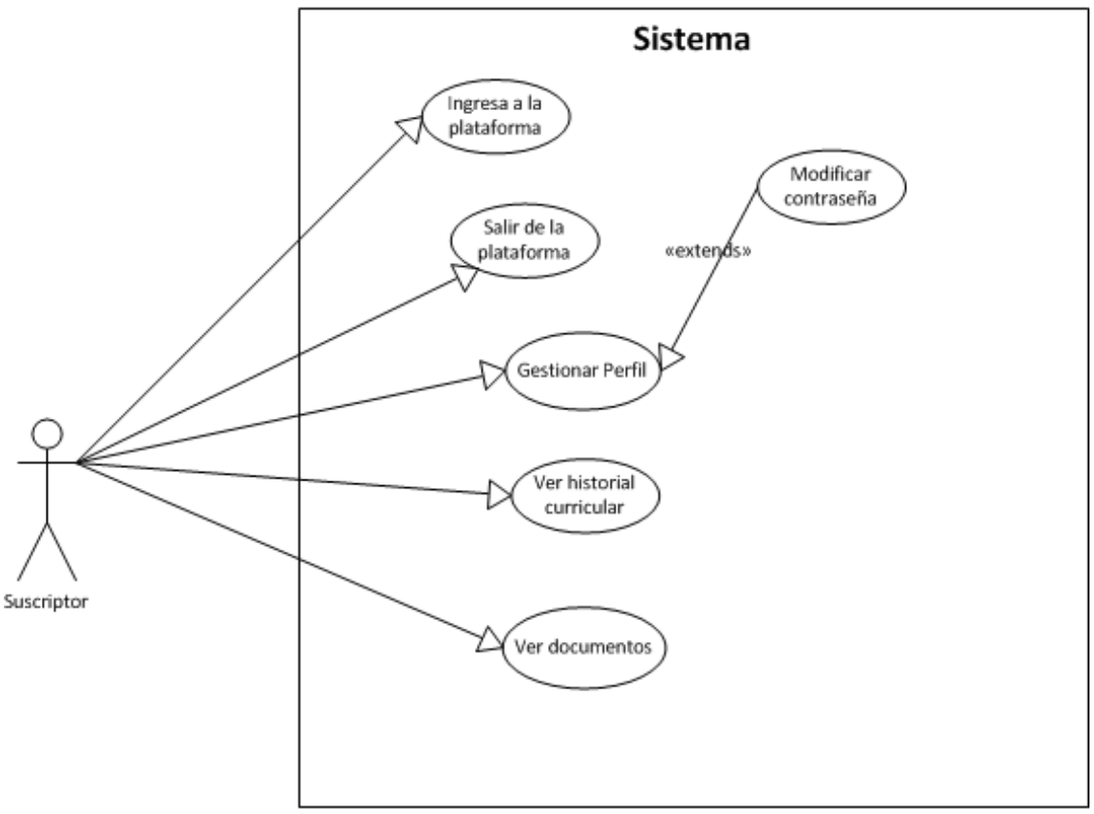


Figura 9. Diagrama de caso de uso para el Suscriptor [13]

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.

[13]Elaboración propia.

3.3. Diseño e implementación

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, y un diagrama de pantallas para ver la interacción del caso de uso con el resto del sistema.

3.3.1. Diagramas de componentes

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 10.

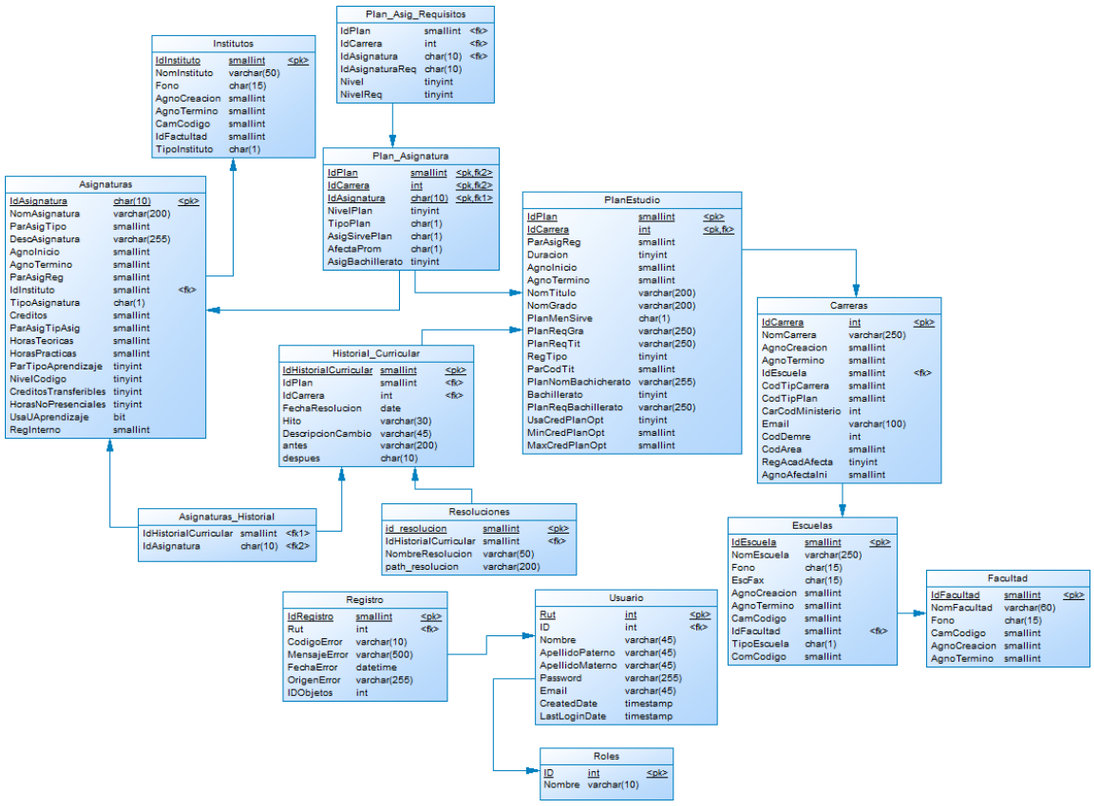


Figura 10. Modelo de datos Entidad-Relación [14]

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.

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.

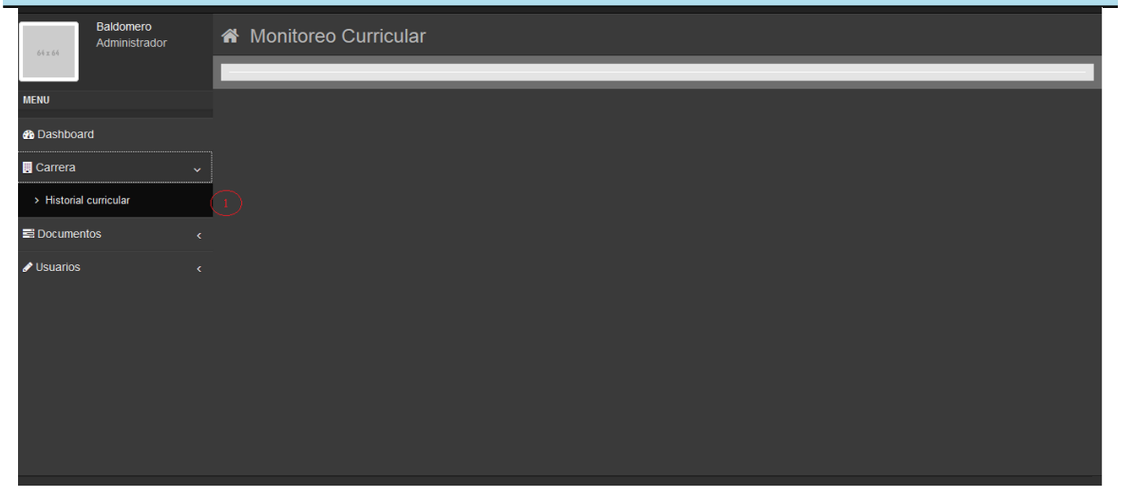
[14] Elaboración propia.

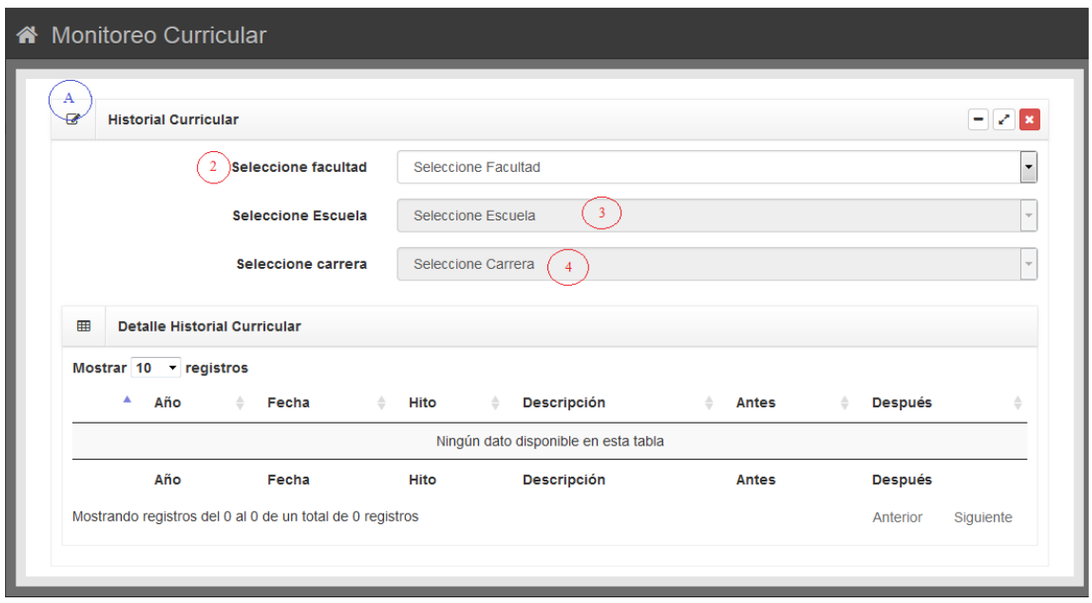
Tabla 3. Caso de uso Ver historial curricular

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 autentificado 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.

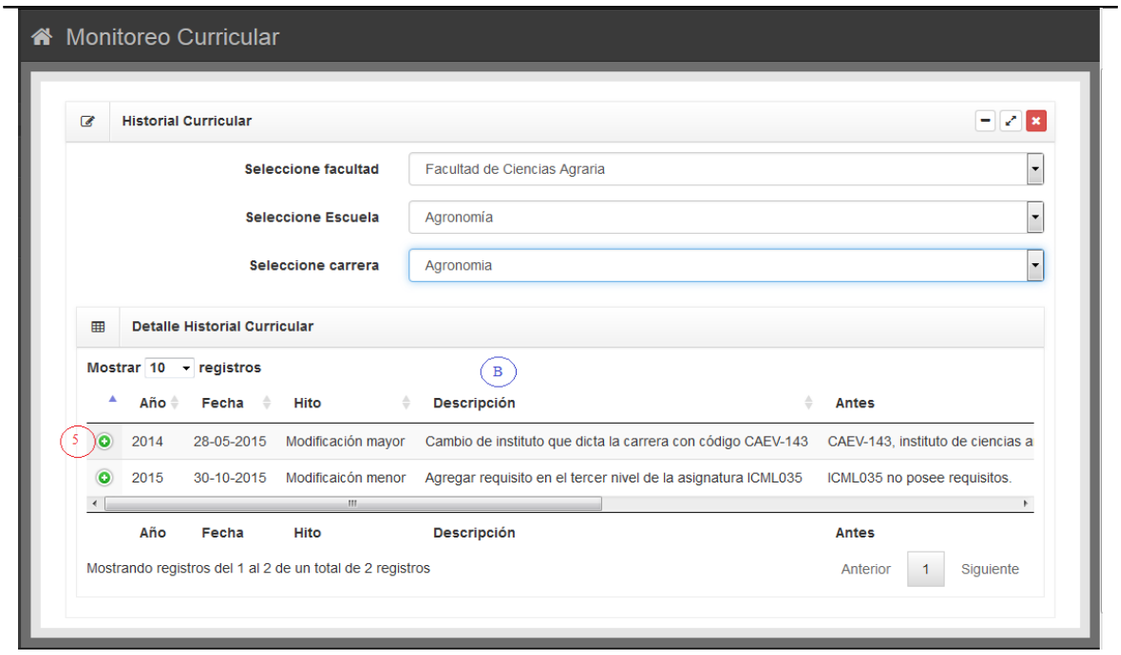
Interfaz: Formulario Ver Historial Curricular

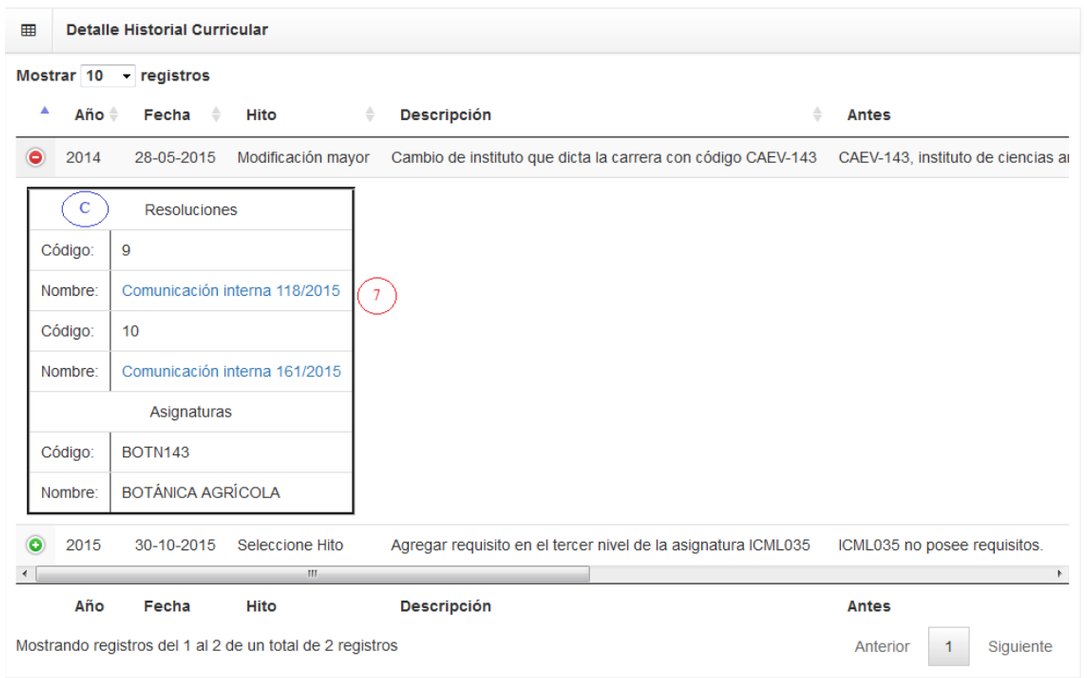




Continúa en la página siguiente

Tabla 3 – *Continuación de la página anterior*





Curso normal de eventos	
Acción actor	**Respuesta sistema**
1.- Este caso de uso comienza cuando un usuario autentificado 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.

Continúa en la página siguiente

Tabla 3 – *Continuación de la página anterior*

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 11 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.

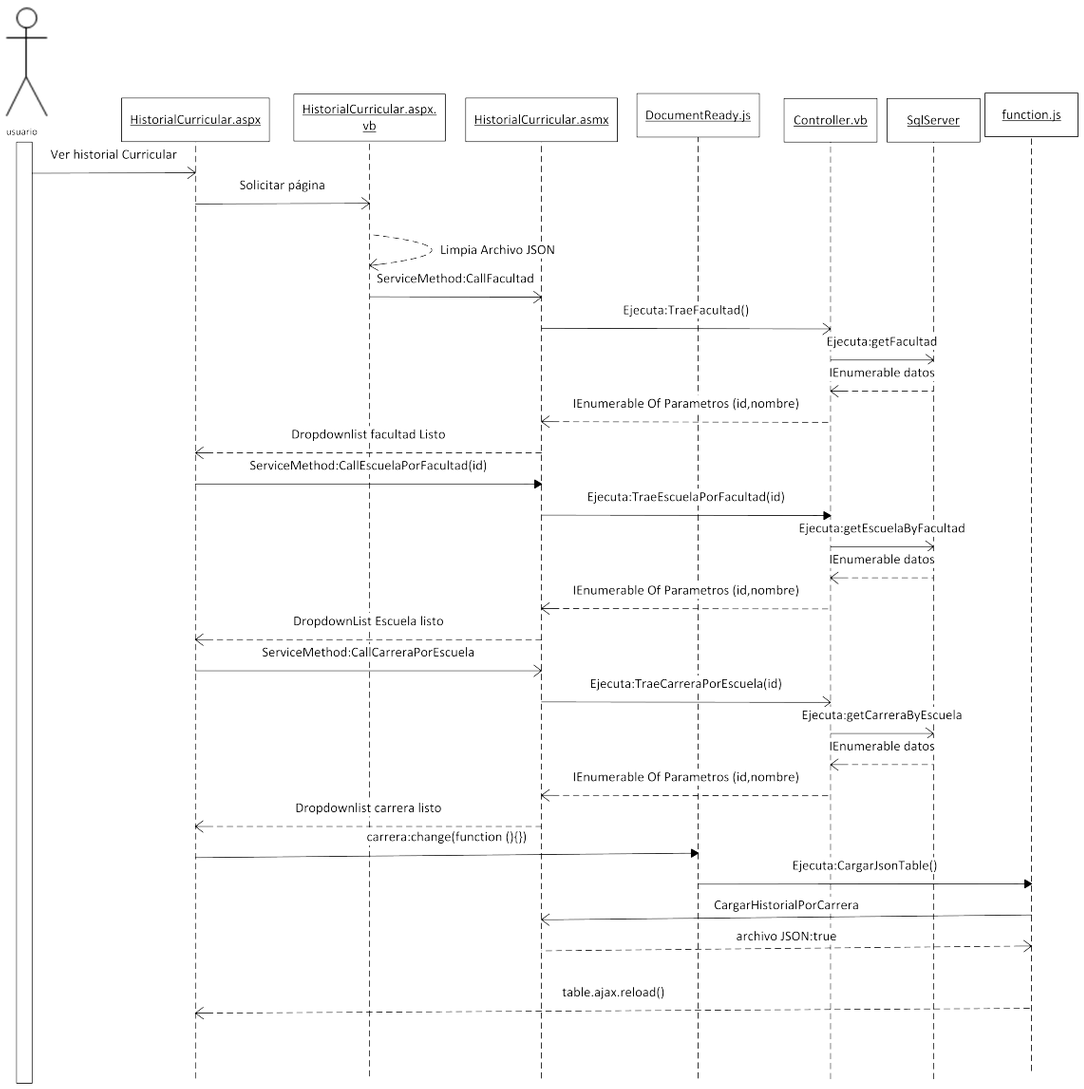


Figura 11. Diagrama de secuencia para el caso de uso *Ver Historial Curricular* [15]

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 arhivo 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

[15]Elaboración 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 proceso que se encarga de poblar una lista desplegable comienza cuando el HistorialCurricular.asmx 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.asmx 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. Metodología

3.4. Validación del software

4. IMPLEMENTACIÓN DEL PROTOTIPO

hola

REFERENCIAS

[MOD07] Universidad Austral de Chile (2007). Modelo educacional y enfoque curricular.

[INN11] Roxana Pey Tumanoff y Sara Chauriye Batarce(2011). INNOVACIÓN CURRICULAR EN LAS UNIVERSIDADES DEL CONSEJO DE RECTORES 2000-2010.

[JQu15] JQuery Disponible en https://jquery.com/. Consultado el 03 de Noviembre de 2015.

[moz15] Mozilla,Firebug Disponible en https://addons.mozilla.org/es/firefox/addon/firebug/. Consultado el 18 de Noviembre de 2015.

[git15] 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.

[eje15] EjemplosTIW Disponible en http://www.lab.inf.uc3m.es/~a0080802/RAI/mvc.html. Consultado el 09 de Noviembre de 2015.

[inf15] Microsoft,Información general sobre ASP.NET Disponible en https://msdn.microsoft.com/es-es/library/dd566231.aspx. Consultado el 10 de Noviembre de 2015.

[GOM10]
Oscar Tinoco Gómez, Pedro Pablo Rosales López & Julio Salas Bacalla,Criterios de selección de metodologías de desarrollo de software 2010 Disponible en http://www.redalyc.org/articulo.oa?id=81619984009. Consultado el 12 de Noviembre de 2015.

[MIC15] 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.

[mic15] Microsoft SQL Server 2008 Express Disponible en https://www.microsoft.com. Consultado el 05 de Noviembre de 2015.

[PAR15] Parsleyjs Disponible en http://parsleyjs.org/doc/about.html. Consultado el 05 de Noviembre de 2015.

[JQu15] JQuery Disponible en https://jquery.com/. Consultado el 03 de Noviembre de 2015.

[gli15] Blog de Glidea, Qué es un template Disponible en http://www.glidea.com.ar/blog/que-es-un-template. Consultado el 04 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.

[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.

[boo15] Bootstrap Disponible en http://getbootstrap.com/about//. Consultado el 04 de Noviembre de 2015.

[ALE15] AlertifyJS Disponible en http://alertifyjs.com/. 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.

[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.

5. ANEXOS

Anexo A: Formato inicial de los archivos JSON.

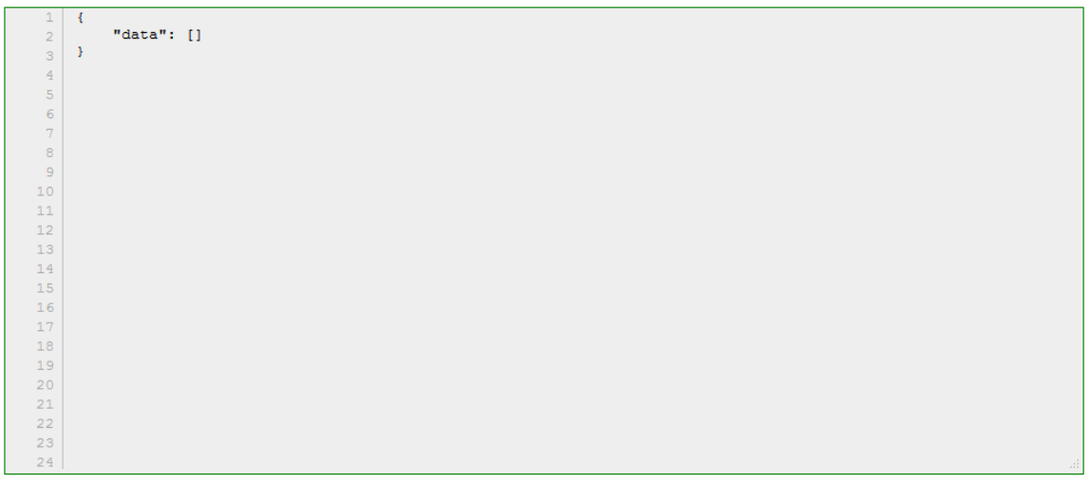


Figura 12. Formato inicial de los archivos JSON.

Anexo B: CascadingDropDown

CascadingDropDown es una extención 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 seleeciona 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.