Instituto Tecnológico Superior de Misantla

Análisis, diseño e implementación del Sistema integral del Instituto Tecnológico Superior de Misantla

TESIS

PARA OBTENER EL GRADO DE:

Maestro en Sistemas Computacionales

Presenta:

Francisco Adolfo Aguilar Gómez

Director:
Jorge Mario Figueroa García

Misantla, Veracruz Agosto de 2018

Autorización de impresión.

En esta sección se agregará el archivo de autorización de impresión.

Agradecimientos.

Quiero expresar mi gratitud sincera a todas aquellas personas que brindaron su apoyo en la realización del presente trabajo, en especial al Dr. Jorge Mario Figueroa García, director de este proyecto, por la orientación, seguimiento y la constante supervisión del mismo, pero sobre todo por la motivación, la paciencia y el gran apoyo brindado durante todo momento.

También agradezco a mis compañeros Carmen Juliana Aguilar Fernández y Ramos Salvador Aquino por esos momentos que pasaron a mi lado en el proyecto así como a mis otros amigos del posgrado por sus palabras de ánimos, su compañerismo, solidaridad, confianza, empatía y amistad. Gracias por todas aquellas experiencias que compartimos.

Dedicatoria.

Dedico esta tesis primeramente a Dios por permitirme tener vida y salud, así como las personas que a continuación menciono porque son lo más precioso que tengo dentro de mi existencia. Esta tesis es por todos ustedes.

A mis padres que con tanto trabajo y sacrificio pudieron brindar una carrera a mi hermana y a mí.

A mi novia Diana Grisel que me ha apoyado en todo este tiempo y siempre fue la mejor compañía que pude tener.

A mi hermana Mercedes por su amistad, confianza y esas palabras de apoyo que mostraron su cariño incondicional.

A mi abuela Julia y mi tía Betty quienes prácticamente me criaron y también amo como si fueran mis madres.

Mi tío Miguel quien se preocupa por mí y aunque no está cerca de forma física sé que puedo contar con él.

Toda mi familia es importante para mí, por ello les agradezco a todos y cada uno de los que la integran, sin ustedes yo no sería la persona de bien en que me he convertido, gracias por su motivación y apoyo.

Resumen.

El Sistema Integral del Tecnológico de Misantla se creó a partir de la necesidad de un nuevo sistema que gestione la información de las actividades claves de las diferentes áreas administrativas, docentes y alumnos en el Instituto Tecnológico Superior de Misantla. Este sistema se divide en diferentes módulos ajustados a los privilegios con los que cuenta cada usuario, pero en esta tesis solo se abarca la creación del módulo de alumnos, es decir su inscripción, sus calificaciones y otros tópicos importantes de su información académica.

En las funciones del módulo alumnos están operando las opciones de inscripción, reinscripción, visualización de kardex y la visualización de historial de adeudos. La plantilla del FrontEnd opera ya satisfactoriamente, se mantiene sujeto a las adecuaciones que requiera el desarrollo en espiral. Con respecto al BackEnd, este se encuentra terminado en su totalidad, superando diversas pruebas de funcionamiento y, además, está operando de manera adecuada en otros proyectos. La capacidad de reutilización del BackEnd y los componentes de la plantilla del FrontEnd pueden ser utilizados en futuros desarrollos de software.

Abstract.

The Sistema Integral del Tecnológico de Misantla was created from the need of a new system that manages the information of the main activities of the different administrative areas, teachers and students in the Higher Technological Institute of Misantla. This system is divided into different modules adjusted to the privileges that each user has, but this thesis only covers the creation of the student module, that is to say, its registration, its qualifications and other important topics of its academic information.

In the module functions students are operating the options of registration, re-enrollment, display of kardex and the display of debt history. The FrontEnd template already operates satisfactorily, it remains subject to the adjustments required by spiral development. With respect to the BackEnd, it is completely finished, overcoming various operational tests and, in addition, it is operating adequately in other projects. The reuse capacity of the BackEnd and the components of the FrontEnd template can be used in future software developments.

Índice general

Autorización de impresión II

Agradecimientos III

Dedicatoria IV

Resumen V

Abstract VI

Índice de figuras X

Índice de tablas XI

Índice de algoritmos XII

1. Generalidades. **13**
1.1. Introducción. . . . 14
1.2. Planteamiento del problema. . . . 15
1.2.1. Justificación. . . . 16
1.3. Objetivos. . . . 16
1.3.1. Objetivo general. . . . 16
1.3.2. Objetivos específicos. . . . 16
1.4. Hipótesis. . . . 17
1.5. Propuesta de solución. . . . 17
1.6. Alcances y limitaciones. . . . 17
1.6.1. Alcances. . . . 17
1.6.2. Limitaciones. . . . 17
1.7. Estructura de la tesis. . . . 18

2. Marco teórico. **19**
2.1. Procesos administrativos en el ITSM. . . . 20
2.1.1. Centro de Idiomas. . . . 20
2.1.2. Centro de Información. . . . 20

2.1.3. Servicios escolares. . . . 21
2.1.4. Área académica. . . . 21
2.2. BackEnd. . . . 21
2.2.1. Base de Datos. . . . 21
2.2.2. PHP. . . . 24
2.3. FrontEnd. . . . 24
2.3.1. CSS. . . . 24
2.3.2. Diseño Web Adaptativo. . . . 25
2.3.3. ECMAScript 6. . . . 25
2.3.4. HTML. . . . 25
2.3.5. Navegador Web. . . . 26
2.4. Programación Orientada a Objetos (POO). . . . 26
2.4.1. Elementos de la POO. . . . 26
2.4.2. Características conceptuales de la POO. . . . 27
2.5. Intercambio de datos. . . . 28
2.5.1. AJAX. . . . 28
2.5.2. JSON. . . . 28
2.6. Patrón de arquitectura de Software. . . . 29
2.6.1. Modelo Vista Control (MVC). . . . 30
2.7. Modelo de procesos del software. . . . 31
2.7.1. Modelo en espiral. . . . 31

3. Estado del arte. 33

4. Metodología. 38

5. Desarrollo de la solución. 40
5.1. Diseño del BackEnd. . . . 41
5.1.1. Gestión de la concurrencia. . . . 42
5.1.2. Gestión del clúster de base de datos heterogéneas y comunicación con los clientes. . . . 43
5.2. Diseño del FrontEnd. . . . 44
5.2.1. Vista. . . . 44
5.2.2. Controlador. . . . 46

6. Resultados. 47
6.0.1. Interfaz de Comunicación. . . . 48
6.0.2. Funciones en el módulo. . . . 49

7. Análisis de resultados. 60
7.1. Comparación entre otras librerías y las librerías desarrolladas. . . . 61
7.2. Análisis de Seguridad Web. . . . 62
7.3. Comparación entre el estado del arte y el sistema ITSM. . . . 63

8. Conclusiones y trabajos futuros. **64**
8.1. Conclusiones. 65
8.2. Trabajos futuros. 65

Referencias **67**

Anexos **70**

Índice de figuras

2.1. Funcionamiento de MVC. 30
2.2. Representación gráfica del modelo en espiral. 32

4.1. Diseño gráfico de la metodología. 39

5.1. Interacción del BackEnd y el FrontEnd. 43
5.2. Adaptación del MVC al proyecto SITM. 44
5.3. Diseño Web Responsivo. 44
5.4. Representación Orientada a Objetos de la información (Tabla, Registro y Campo). 45
5.5. Representación Orientada a Objetos de la información (bloque, materias, materia). 46

6.1. Diagrama descriptivo de la plantilla. 48
6.2. Materias disponibles. 49
6.3. Diagrama de secuencia de la inscripción del alumno. 50
6.4. Vista previa de horario. 51
6.5. Vista Kardex. 51
6.6. Vista adeudos. 52
6.7. Diagrama de flujo para usar el kardex (1ra parte). 57
6.8. Diagrama de flujo para usar el kardex (2da parte). 58
6.9. Diagrama de flujo para usar el kardex (3ra parte). 59

Índice de tablas

7.1. Tabla comparativa de librerias. 61
7.2. Tabla comparativa con otros sistemas implementados en otros tecnológicos. . 62
7.3. Tabla comparativa con otros sistemas implementados en el estado del arte. . 63

Índice de algoritmos

5.1. Singleton para el manejo de la conexion a MSSQL. 42
8.1. Código de index.html. 70
8.2. Código CSS3 de la plantilla 71
8.3. Código de la clase mensajes. 78
8.4. Código de la clase tabla. 80
8.5. Código de la clase registro. 81
8.6. Código de la clase campo. 81
8.7. Código de la clase kardex. 81
8.8. Código de la plantilla SITM. 85
8.9. Código del index. 90
8.10. Código de modulo_ev. 93

Capítulo 1

Generalidades.

1.1. Introducción.

La planeación, organización y dirección escolar son actividades necesarias en una institución educativa, pero la complejidad para la gestión de ellas se incrementa dependiendo del nivel educativo. En el mercado existe software disponible de diferentes desarrolladores para gestionar las actividades academicas, pero no cumplen todas las necesidades actuales de instituciones de educación superior. La presente tesis realiza el análisis y seguimiento del desarrollo de un Enterprise Resource Planning (ERP) utilizando diferentes tecnologías estandarizadas de desarrollo web. En su desarrollo se usan los principios de la Programación Orientada a Objetos (POO) con la finalidad de implementar la reutilización de componentes de software en los códigos generadores de la presentación y la transferencia de la información entre los clientes y el servidor. Este proyecto está destinado tanto para el área académica en el Instituto Tecnológico Superior de Misantla (ITSM) asi como sus docentes y alumnos, pero en este documento solo se presentara el desarrollo del módulo "alumnos", en este módulo el alumno podrá hacer uso de las funciones principales de inscripción, reinscripción, visualización de kardex y la visualización de su historial de adeudos, todo lo anterior a través de un dispositivo con conexión a internet.

1.2. Planteamiento del problema.

En el ITSM se cuenta actualmente con un sistema monolítico poco flexible para la gestión de los servicios escolares y jefaturas de carrera, el cual es considerado desfasado para las necesidades actuales que tiene la institución, carente además de capacidades de interconectividad entre sistemas e imposible de modificar para agregar nuevos módulos. Por estrategia, el ITSM apuesta hacia el uso y explotación de las Tecnologías de la Información y las Comunicaciones en la gestión e impartición de carreras que en su interior se brindan. Tras un análisis de la parte directiva con la parte operativa se concluyó que: es necesario contar con un sistema de gestión que integre la parte académica, administrativa, operativa y directiva en sus diferentes niveles y que sea capaz de interconectar con otros sistemas heterogéneos. Un ejemplo de los problemas con el sistema actual es en la reinscripción de un alumno. El departamento de servicios escolares debe cerciorarse que el solicitante no tenga adeudos. Los adeudos pueden ser un pago, un libro o una actividad requerida por el plan de estudios, todos estos adeudos son registrados en otra base de datos sin conexión a la de servicios escolares, así que si el alumno por ejemplo, realiza un pago en caja pero servicios escolares no registra este pago en su propia base de datos, el joven queda como deudor cuando realmente él ya ha pagado su cuenta en caja. Para registrar los adeudos, servicios escolares actualizan su base de datos cada semestre de forma manual a partir de una lista impresa, la cual fue extraída de la base de datos en otro departamento, esta es una actividad extra para servicios escolares, además hace necesario rectificar muchas veces su propia información y este proceso se realiza por cada alumno inscrito. Otro ejemplo es la asignación de horarios de los docentes. Los jefes de carrera fijan los horarios de cada grupo, si se trata de un grupo regular el horario se asigna con sucesión ininterrumpida de horas, pero hay casos en que el grupo no es regular y hace necesario abrir un curso que está marcado fuera de la planeación escolar, estos casos especiales son muy recurrentes y hacen estragos en las actividades haciendo más difícil crear una carga académica por grupo. Pero no solo el problema termina en el grupo, también hay alumnos que no son regulares y los cuales deben armar de forma individual sus horarios sin negarles los cursos marcados por el plan de estudios con los pocos docentes compartidos entre carreras en la institución. Estos y muchos más son los problemas que el jefe de carrera debe de resolver al crear los horarios.

1.2.1. Justificación.

El ITSM requiere un nuevo sistema con nuevas funciones que satisfaga las diferentes necesidades de cada uno de sus departamentos, docentes y alumnos. Existen diferentes sistemas en el mercado pero ninguno se adecua a las necesidades de la institución, en la mayoría de los casos es necesario comprar una licencia de uso para estas opciones. Tras la consideración del cuerpo directivo y el departamento de sistemas del ITSM se tomó la decisión de implementar e implantar un ERP que satisfaga las necesidades operativas y administrativas de carácter primario con la posibilidad de ser adecuado a otras nuevas en cualquier momento, que este abierto a conexión con otras plataformas y que pueda ser ampliado cuando sea oportuno.

Con el Sistema Integral del Tecnológico de Misantla (SITM) se satisfará la necesidad de automatización de los procesos administrativos y docentes de carácter primario, lo que permitirá al personal administrativo, docente y alumnos contar con información oportuna que respalde la toma de decisiones. Con respecto al desarrollo técnico de este sistema se optó por crear una plantilla programada en JavaScript completamente independiente de librerías de terceros, con un enfoque orientado a objetos y un diseño adaptativo capaz de adaptarse al tamaño de cualquier dispositivo, esta plantilla es capaz de reutilizarse en otros proyectos distintos en un futuro.

1.3. Objetivos.

1.3.1. Objetivo general.

Analizar, diseñar e implementar un sistema que respalde las necesidades académicas, administrativas, directivas y de operación del ITSM.

1.3.2. Objetivos específicos.

- Diseñar e implementar una estructura estandarizada para todos los módulos en el SITM que funcione como plantilla de presentación para el usuario.
- Diseñar e implementar el módulo alumnos, además, su interconectividad con el BackEnd y los módulos en desarrollo del sistema.
- Validar y analizar resultados.
- Comparar los resultados con los resultados del estado del arte.

1.4. Hipótesis.

Con la implementación de este sistema mejorará el proceso actual de gestión de información académica y administrativa que se realizan en el ITSM, específicamente: inscripción y reinscripción de alumnos así como la asignación de horario de materias.

1.5. Propuesta de solución.

El SITM es un ERP sobre una plataforma web para sistematizar, integrar y automatizar los procesos primordiales del Tecnológico de Misantla. En su desarrollo se usan los principios de la POO para lograr la reutilización de componentes de software en los códigos generadores de la presentación para el usuario, además de la transferencia de la información entre los clientes y el servidor resultando en: un modelo de comunicación muy simple, transferencia de códigos y datos reducida al mínimo, una eficiente generación visual de los componentes de interfaz y de forma inherente se ha elevado la seguridad del sistema. Logrando con ello la creación de una herramienta de alto nivel para el área académica y administrativa al alcance de todos los involucrados anexando funciones al sistema de coordinación utilizado hasta este momento en el ITSM e instituciones similares.

1.6. Alcances y limitaciones.

1.6.1. Alcances.

Para el módulo alumnos se desarrollaran las siguientes funciones:

- Visualización del kardex del alumno.
- Asignación de Carga académica del alumno.
- Actualización del registro de kardex con la asignación de la carga académica del alumno.

1.6.2. Limitaciones.

- Escasa información existente de trabajos relacionados con este proyecto.
- Acceso limitado a información de carácter confidencial en la institución.

1.7. Estructura de la tesis.

El contenido del desarrollo se ha dividido en los siguientes capítulos:

1. En el primer capítulo se plantea la problemática así como también se establecen la justificación, los objetivos, la hipótesis, la propuesta de solución, los alcances y limitaciones de este proyecto.

2. El segundo capítulo refiere al marco teórico, este muestra la importancia de las tecnologías en el desarrollo de software en el ambiente web.

3. El tercer capítulo pertenece al estado del arte, en este se muestra algunos artículos de distintos proyectos relacionados con esta tesis de manera directa o indirecta a partir de sus resultados, metodologías o implementación.

4. En el cuarto capítulo está el desarrollo de la solución y muestra la metodología elegida así el proceso de diseño y creación.

5. El quinto capítulo muestra el análisis de resultados.

6. El sexto capítulo son las conclusiones y los trabajos futuros que se tiene a partir del resultado de este trabajo.

Capítulo 2

Marco teórico.

Para una mejor comprensión de este trabajo es menester plantear algunos conceptos que forman los pilares principales en el desarrollo de este proyecto, estos se presentan a continuación:

2.1. Procesos administrativos en el ITSM.

Actualmente el Tecnológico Superior de Misantla cuenta con diversos departamentos administrativos los cuales cubren sus necesidades de registro por medios distintos, a continuación se muestran dichos departamentos y una explicación de sus funciones.

2.1.1. Centro de Idiomas.

El centro de idiomas cuenta con un sistema a medida desarrollado por la Mtra. Elsa Irene Santiago en el lenguaje PHP. Entre las funciones de este sistema están: El alta de un alumno al nivel que tomara y la asignación de su horario, calificación final del módulo, registro del pago realizado en caja por el alumno, consulta de historial del alumno. Entre los problemas que tiene el sistema esta: La creación de la hoja de acreditación, duplicidad de Información, falta de consultas, modificación de la información de los pagos que se ingresaron.

2.1.2. Centro de Información.

El Centro de información del ITSM cuenta con la versión 4 del Sistema Integral Automatizado de Bibliotecas de la Universidad de Colima (SIABUC), fue adquirido directamente en la universidad de colima en el año 2005 y es usado en el área desde entonces, los módulos que están habilitados para el personal del Centro de Información son:

- Consulta de Libros: En este módulo se puede realizar la búsqueda de cualquier libro que se haya dado de alta en el sistema, esta se puede efectuar ya sea por el nombre del libro o por autor.
- Préstamos: En este módulo se registran los préstamos que se realizan a diario ya sea para consulta interna o externa.
- Análisis: Dentro de este módulo el personal que opera el sistema puede descargar información que es requerida para fines estadísticos.
- Respaldo: Aquí se realiza el respaldo de la base de datos, para poder tener un resguardo de la información que se alberga en ella.

La problemática de este sistema es que para fines estadísticos solo se puede consultar información general y no como la solicita el área directiva, por lo tanto, para realizar esta tarea se descarga la información y se clasifica en macros de Excel. No existe interconexión con la base de datos del departamento de Servicios Escolares para tener actualizada la información de alumnos.

2.1.3. Servicios escolares.

En servicios escolares se utiliza desde el año 2000 el Sistema de Integración Escolar (SIE), es un sistema administrativo y de control escolar desarrollado por el Instituto Tecnológico Superior de Xalapa después de varios intentos fallidos de la Dirección General de Educación Superior Tecnológica (DGEST) por desarrollar un sistema administrativo para todos los institutos tecnológicos. El SIE en niveles totales genera registros de: matricula, egresos, aspirantes a ingreso, alumnos con becas, profesores, servicio social, proyectos de investigación, convenios de vinculación, personal docente y no docente. También genera reportes e impresión de documentos de carácter oficial como el Kardex y la boleta de calificaciones. Entre las limitaciones del SIE destaca la falta de registro del proceso de titulación, seguimiento de Residencias profesionales y las estadísticas de la población estudiantil.

2.1.4. Área académica.

El área académica no cuenta con un sistema que tenga como propósito el cubrir sus necesidades, actualmente los registros son capturados a través de formatos previamente desarrollados en Microsoft Access, dichos formatos son para registrar las adscripciones del personal docente, en las cuales se especifican las horas de trabajo que debe cubrir, asi como las actividades que realiza en las horas: frente a grupo, complementarias y de bloque (Semestre). Aunque actualmente están cubiertas las necesidades del área, la plantilla de registro no cuenta con una interfaz propia.

2.2. BackEnd.

Define (Caballero, 2016) a éste como: “las actividades realizadas del lado del servidor; es decir, las tareas de base de datos y los servidores de aplicaciones que el usuario no puede visualizar en el explorador de Internet. Los lenguajes usados comúnmente son PHP, Java, Ruby, .NET, ASP, Python, entre otros, los cuales son los encargados de interactuar con la base de datos”.

2.2.1. Base de Datos.

En la obra (de Guevara, 2018) se afirma que: “cada día, la mayoría de nosotros nos encontramos con actividades que requieren algún tipo de interacción con una base de datos (ingreso en un banco, reserva de una entrada para el teatro, solicitud de una suscripción a una revista, compra de productos, ...). Estas interacciones son ejemplos de lo que se llama aplicaciones tradicionales de bases de datos (básicamente información numérica o de texto), aunque los avances tecnológicos han permitido que también existan: bases de datos multimedia, sistemas de información geográfica (GIS), almacenes de datos, sistemas de proceso analítico on-line, ...”.

- Una base de datos se entenderá como una colección de datos relacionados entre sí y que tienen un significado implícito.
- Por datos queremos decir hechos conocidos que pueden registrarse y que tienen un significado implícito.

La definición presentada anteriormente hace referencia a dos elementos para que un conjunto de datos constituya una Base de Datos:

1. Relaciones entre datos.
2. Significado implícito.

Normalización.

Normalizar las tablas de una base de datos (del Carmen Gómez Fuentes, 2013) dice que significa optimizar su diseño para no repetir datos innecesariamente y para prevenir problemas de inconsistencia. Con la normalización, los datos complejos se transforman en un conjunto de tablas simples que son más fáciles de entender y mantener. Es mejor invertir un esfuerzo importante al diseño de la base de datos, esto ahorra mucho esfuerzo en las etapas posteriores del desarrollo de un sistema de software.

E. F. Codd, fue el creador de las bases de datos relacionales, en 1970 definió las tres primeras formas normales, que son:

1FN (primera Forma Normal) E. F. Codd, fue el creador de las bases de datos relacionales, en 1970 definió las tres primeras formas normales, que son:

- **La primera forma normal (1NF):** Una tabla se encuentra en primera forma normal si y sólo si los valores que componen el atributo de una tupla son atómicos. Los atributos atómicos, son valores únicos y lo más pequeño posible, es decir, indivisibles. Además tiene una clave primaria, atributos no nulos y no tiene tuplas repetidas.
- **La segunda forma normal (2NF):** Una tabla está en la segunda forma normal cuando está en la primera forma normal (1NF) y además cada campo secundario (aquel que no pertenece a la clave principal) depende de la clave principal en su totalidad y no de una parte de ella.
- **La tercera forma normal (3NF):** Una tabla está en la tercera forma normal cuando esta en la segunda forma normal (2NF) y además cada campo que no sea llave primaria solo depende de la llave primaria o de las claves secundarias de la tabla y no depende de otro campo de tal forma que "no existen atributos no primarios que son transitivamente dependientes de cada posible clave de la tabla".

Se dice que una tabla cumple una determinada forma normal cuando satisface las restricciones impuestas por dicha norma. La idea es evitar los problemas que surgen cuando una base de datos está mal diseñada.

Durante el proceso de normalización hay que tener siempre en mente la posibilidad e descomponer una tabla en otras más pequeñas.

La normalización tiene como ventajas principales:

- La precisión.- Las interrelaciones entre las tablas consiguen mantener información diferente relacionada con toda exactitud.
- La mínima redundancia.- la información no está duplicada innecesariamente.

Sistema Gestor de Bases de Datos.

También (del Carmen Gómez Fuentes, 2013) nos define un sistema gestor de bases de datos (SGBD) como "una aplicación que permite a los usuarios definir, crear y mantener una base de datos, y proporciona acceso controlado a la misma". Además establece que en general, un SGBD proporciona los siguientes servicios:

- Permite la **definición de la base de datos** mediante el lenguaje de definición de datos (**DDL – Data Description Language**). Este lenguaje permite especificar la estructura y el tipo de los datos, así como las restricciones sobre los datos. Todo esto se almacenará en la base de datos.
- Permite la **inserción, actualización, eliminación y consulta de datos** mediante el lenguaje de manejo o manipulación de datos (**DML - Data Manipulation Language**).
- Proporciona un acceso controlado a la base de datos mediante:
  - Un sistema de seguridad, de modo que los usuarios no autorizados no puedan acceder a la base de datos, mediante el lenguaje de control de datos (DCL - Data Control Language).
  - Un sistema de integridad que mantiene la integridad y la consistencia de los datos.
  - Un sistema de control de concurrencia que permite el acceso compartido a la base de datos.
  - Un sistema de control de recuperación que restablece la base de datos después de que se produzca un fallo del hardware o del software.
  - Un **diccionario de datos** o catálogo accesible por el usuario que contiene la descripción de los datos de la base de datos.

La principal herramienta de un SGBD es la interfaz de programación con el usuario. Esta interfaz consiste en un lenguaje muy sencillo mediante el cual el usuario interactua con el servidor. Este lenguaje comúnmente se denomina **SQL, Structure Query Language**, está estandarizado por la ISO 1, es decir, todas las BD que soporten SQL deben tener la misma sintaxis a la hora de aplicar el lenguaje.

2.2.2. PHP.

La página web oficial (php.net, 2018) dice que "PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. En lugar de usar muchos comandos para mostrar HTML (como en C o en Perl), las páginas de PHP contienen HTML con código incrustado que hace 'algo'. El código de PHP está encerrado entre las etiquetas especiales de comienzo y final <?php y ?> que permiten entrar y salir del 'modo PHP'. Lo que distingue a PHP del lado del cliente como Javascript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá el resultado de ejecutar el script, aunque no se sabrá el código subyacente que era. El servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP, por lo que no hay manera de que los usuarios puedan saber qué se tiene debajo de la manga. Lo mejor de utilizar PHP es su extrema simplicidad para el principiante, pero a su vez ofrece muchas características avanzadas para los programadores profesionales".

2.3. FrontEnd.

Tambien según (Caballero, 2016) "dentro del contexto del desarrollo de aplicaciones web, implica el uso de las tecnologías con las que interactúa directamente el usuario. Normalmente estas tecnologías son desarrolladas en los lenguajes de HTML, CSS y Javacript; también se usan las herramientas de diseño gráfico como Photoshop o Fireworks. El objetivo es desarrollar la interfaz gráfica de usuario (GUI), buscando una experiencia de uso bien valorada por el usuario final, siendo en algunos casos necesario hacer investigación, estudios y pruebas para llegar a este fin. Además, dentro del desarrollo de las aplicaciones web es posible desarrollar el front-end de la aplicación sin contar con una aplicación back-end que interactúe con la base de datos".

2.3.1. CSS.

"Hojas de Estilo en Cascada (Cascading Style Sheets), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos" definición oficial de (W3C, 2016).

2.3.2. Diseño Web Adaptativo.

"Este término es usado en referencia a la manera en que cada explorador de Internet responde a su ambiente, aportando así a la mejora de la experiencia del usuario en el uso del sitio, independientemente del tamaño de resolución y pantalla del dispositivo. Para lograr esto es imprescindible el uso de CSS3 acompañado del uso media queries dentro del mismo CSS", (Caballero, 2016).

Según (Marcotte, 2011) existen tres elementos fundamentales para lograr un diseño web adaptativo, que son:

- Cuadrícula fluida: Este concepto lo que propone es usar porcentajes para definir los tamaños de las columnas o divs, en lugar de píxeles.
- Imágenes flexibles: Las imágenes no tienen anchos fijos sino un máximo (max-width), que en una computadora de escritorio o laptop suele mostrarse al 100 % y se ajustara al tamaño de su contenedor si este reduce su tamaño.
- Media queries: Con el uso de estilos CSS permite que se personalice utilizando un ancho mínimo y máximo del navegador (min-max width).

2.3.3. ECMAScript 6.

"ECMAScript 6 es un lenguaje de programación orientado a objetos para realizar cálculos y manipular objetos computacionales dentro de un entorno host. ECMAScript como se define aquí no pretende ser computacionalmente autosuficiente; de hecho, no hay disposiciones en esta especificación para la entrada de datos externos o salida de resultados calculados. En su lugar, se espera que el entorno computacional de un programa ECMAScript proporcione no solo los objetos y otras instalaciones descritas en esta especificación sino también ciertos objetos específicos del entorno, cuya descripción y comportamiento están más allá del alcance de esta especificación excepto para indicar que puede proporcionar ciertas propiedades a las que se puede acceder y ciertas funciones que se pueden invocar desde un programa ECMAScript", definición oficial de (ecma international.org, 2015).

2.3.4. HTML.

El lenguaje de marcado de hipertexto (the Hypertext Markup Language) es de forma oficial según el (W3C, 2016) "el lenguaje para describir la estructura de las páginas web". HTML da a los autores los medios para:

- Publicar documentos en línea con encabezados, texto, tablas, listas, fotos, etc.
- Recuperar información en línea a través de enlaces de hipertexto, con solo presionar un botón.

- Diseñar formularios para realizar transacciones con servicios remotos, para usar en la búsqueda de información, realizar reservas, solicitar productos, etc.
- Incluir hojas de cálculo, videoclips, clips de sonido y otras aplicaciones directamente en sus documentos.

Con HTML, los autores describen la estructura de las páginas usando el marcado. Los elementos de la etiqueta de idioma incluyen fragmentos de contenido como 'párrafo', 'lista', 'tabla', etc.

2.3.5. Navegador Web.

Se explica en (Valzacchi, 2003) que: "una arquitectura de tipo cliente-servidor, el usuario interactúa y obtiene información desde su computadora a través de una aplicación cliente. En la Web estas aplicaciones se conocen bajo el nombre genérico de 'browsers' (también llamadas en nuestro idioma 'visores', 'visualizadores', 'navegadores' ó 'exploradores')", y cumplen dos funciones básicas:

1. Transmitir a los servidores remotos las órdenes que le imparte el usuario.
2. Presentar la información en forma asequible a quien la solicite.

Los navegadores Web más utilizados por volumen de uso actualmente son: Google Chrome, Internet Explorer, Mozilla Firefox, Safari y Opera.

2.4. Programación Orientada a Objetos (POO).

"La POO es un paradigma de programación (o técnica de programación) que utiliza objetos e interacciones en el diseño de un sistema", dice (Bahit, 2011). También dice que tiene sus propias caracteríscas y elementos que se mencionan a continuación:

2.4.1. Elementos de la POO.

La POO está compuesta por una serie de elementos que se detallan a continuación.

- **Clase**: Una clase es un modelo que se utiliza para crear objetos que comparten un mismo comportamiento, estado e identidad.
- **Objeto:** Es una entidad provista de métodos o mensajes a los cuales responde (comportamiento); atributos con valores concretos (estado); y propiedades (identidad).
- **Método:** Es el algoritmo asociado a un objeto que indica la capacidad de lo que éste puede hacer.

- **Evento y Mensaje:** Un evento es un suceso en el sistema mientras que un mensaje es la comunicación del suceso dirigida al objeto.
- **Propiedades y atributos:** Las propiedades y atributos, son variables que contienen datos asociados a un objeto.

2.4.2. Características conceptuales de la POO.

La POO debe guardar ciertas características que la identifican y diferencian de otros paradigmas de programación. Dichas características se describen a continuación.

- **Abstracción:** Aislación de un elemento de su contexto. Define las características esenciales de un objeto
- **Encapsulamiento:** Reúne al mismo nivel de abstracción, a todos los elementos que puedan considerarse pertenecientes a una misma entidad.
- **Modularidad:** Característica que permite dividir una aplicación en varias partes más pequeñas (denominadas módulos), independientes unas de otras.
- **Ocultación (aislamiento):** Los objetos están aislados del exterior, protegiendo a sus propiedades para no ser modificadas por aquellos que no tengan derecho a acceder a las mismas.
- **Polimorfismo:** Es la capacidad que da a diferentes objetos, la posibilidad de contar con métodos, propiedades y atributos de igual nombre, sin que los de un objeto interfieran con el de otro.
- **Herencia:** Es la relación existente entre dos o más clases, donde una es la principal (madre) y otras son secundarias y dependen (heredan) de ellas (clases "hijas"), donde a la vez, los objetos heredan las características de los objetos de los cuales heredan.
- **Recolección de basura:** Es la técnica que consiste en destruir aquellos objetos cuando ya no son necesarios, liberándolos de la memoria.

2.5. Intercambio de datos.

Es la transmisión estructurada de datos por medios electrónicos. Se usa para transferir datos de un sistema computacional a otro. Este intercambio puede realizarse en distintos formatos y transferirse a través de distintos métodos.

2.5.1. AJAX.

"JavaScript Asíncrono y XML (AJAX) no es una tecnología por sí misma, es un término que describe un nuevo modo de utilizar conjuntamente varias tecnologías existentes. Esto incluye: HTML o XHTML, CSS, JavaScript, DOM, XML, XSLT, y el objeto XMLHttpRequest. Cuando estas tecnologías se combinan en un modelo AJAX, es posible lograr aplicaciones web capaces de actualizarse continuamente sin tener que volver a cargar la página completa. Esto crea aplicaciones más rápidas y con mejor respuesta a las acciones del usuario". Definición oficial de (mozilla, 2012).

2.5.2. JSON.

"JavaScript Object Notation - Notación de Objetos de JavaScript es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos".

JSON está constituído por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o sequencias.

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras". Definición oficial de (Json.org, 2013).

2.6. Patrón de arquitectura de Software.

Indica (Johanna M. Suárez, 2015) que: "La Arquitectura de Software, según el estándar IEEE 1471-2000 (IEEE, 2001), se define como la organización fundamental de un sistema incorporando sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución. Un patrón de arquitectura busca solucionar problemas de adaptabilidad de requerimientos, rendimiento, modularidad y acoplamiento de los componentes de una aplicación. Los patrones de arquitectura, dentro o entre los niveles arquitectónicos, están relacionados con la interacción de los objetos". Tambien dice que "Los patrones de arquitectura, al igual que los patrones de diseño, representan llamadas entre objetos, decisiones y criterios arquitectónicos, así como la manera de empaquetado de funcionalidades de una aplicación. Sin embargo, los patrones de arquitectura representan un nivel más alto en el sistema.

Este tipo de patrones proveen un conjunto de subsistemas predefinidos, especifican las responsabilidades e incluyen reglas y guías para organizar las relaciones entre estos subsistemas". Algunos patrones de arquitectura son:

- Patrón Layers.
- Patrón Broker.
- Patrón MVC.

El MVC es el patrón de arquitectura que describe el presente proyecto.

2.6.1. Modelo Vista Control (MVC).

La definición de (Castejón, 2004) sobre el MVC es: "este patrón propone la separación en distintos componentes de la interfaz de usuario (vistas), el modelo de negocio y la lógica de control. Una vista es una 'fotografía' del modelo (o una parte del mismo) en un determinado momento. Un control recibe un evento disparado por el usuario a través de la interfaz, accede al modelo de manera adecuada a la acción realizada, y presenta en una nueva vista el resultado de dicha acción. Por su parte, el modelo consiste en el conjunto de objetos que modelan los procesos de negocio que se realizan a través del sistema.
En una aplicación web, las vistas serían las páginas HTML que el usuario visualiza en el navegador. A través de estas páginas el usuario interactúa con la aplicación, enviando eventos al servidor a través de peticiones HTTP. En el servidor se encuentra el código de control para estos eventos, que en función del evento concreto actúa sobre el modelo convenientemente. Los resultados de la acción se devuelven al usuario en forma de página HTML mediante la respuesta HTTP". (Bahit, 2011) describe de forma gráfica el funcionamiento básico del patrón MVC en la Figura 2.1.

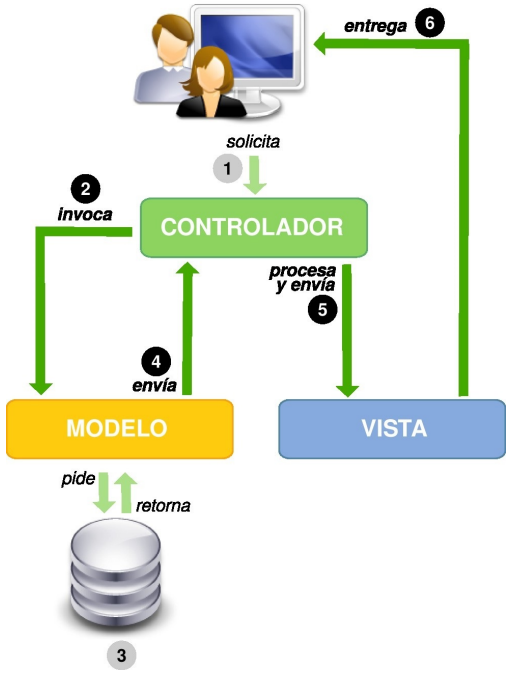

Figura 2.1: Funcionamiento de MVC.

2.7. Modelo de procesos del software.

"Son una descripción simplificada de un proceso del software que presenta la visión del proceso. Estos modelos pueden incluir actividades que son parte de los procesos y productos de software y el papel de las personas involucradas en la ingeniería de software", afirma (Sommerville, 2005). Algunos ejemplos de estos tipos de modelos que se pueden producir son:

- El enfoque en cascada.
- Desarrollo iterativo.
- Ingeniería del software basada en componentes (CBSE).

A continuación se describe el modelo de desarrollo que se utilizó para este proyecto.

2.7.1. Modelo en espiral.

"Es un enfoque realista para el desarrollo de sistemas y de software a gran escala. Como el software evoluciona a medida que el proceso avanza, el desarrollador y cliente comprenden y reaccionan mejor ante los riesgos en cada nivel de evolución. El modelo espiral usa los prototipos como mecanismo de reducción de riesgos, pero, más importante, permite aplicar el enfoque de hacer prototipos en cualquier etapa de la evolución del producto. Mantiene el enfoque de escalón sistemático sugerido por el ciclo de vida clásico, pero lo incorpora en una estructura iterativa que refleja al mundo real en una forma más realista. El modelo espiral demanda una consideración directa de los riesgos técnicos en todas las etapas del proyecto y, si se aplica de manera apropiada, debe reducir los riesgos antes de que se vuelvan un problema" dice (Pressman, 2010). Se presenta en la Figura 2.1 el modelo en espiral del proceso de software propuesto por (Boehm, 1988).

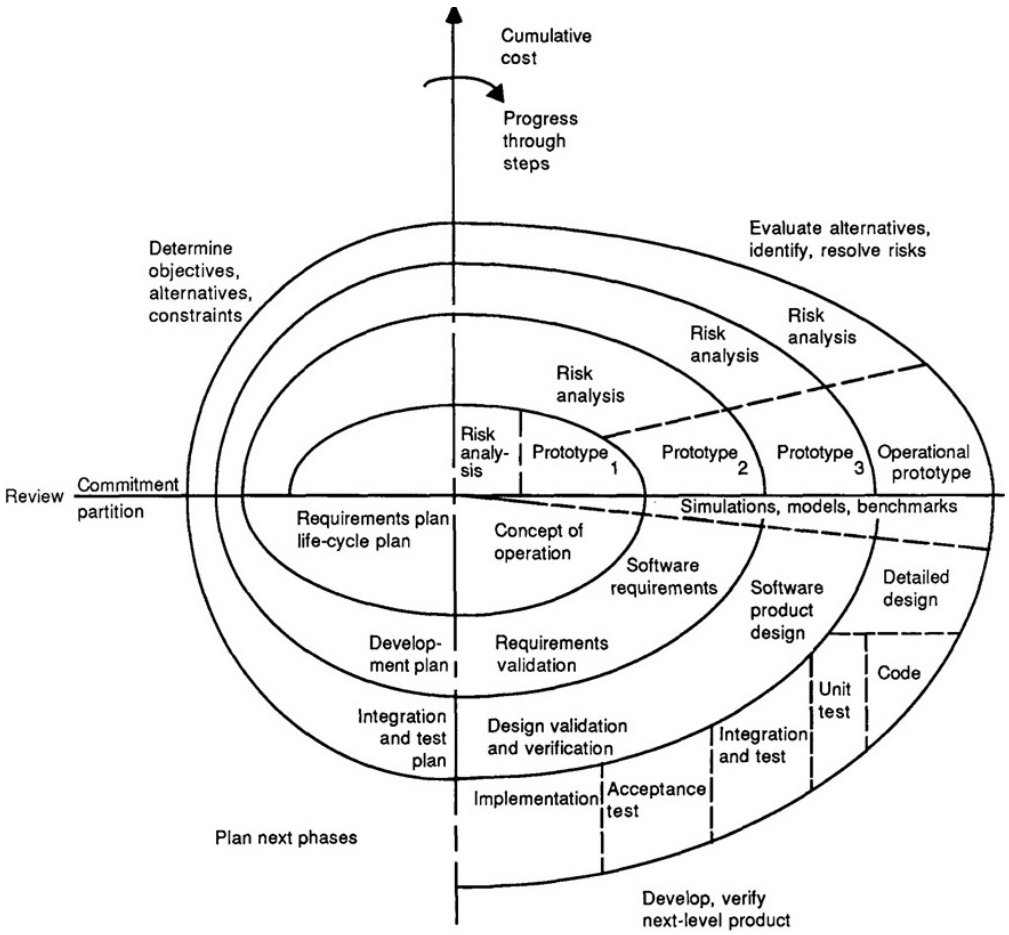

Figura 2.2: Representación gráfica del modelo en espiral.

Capítulo 3

Estado del arte.

Rivera-Silva *et al.*, en su artículo "Implementación de recursos empresariales (ERP) en las organizaciones desde la co-evolución" analizan a través de la literatura la implementación de ERP en distintas organizaciones e identifican que el 70 % de las empresas no obtuvieron los resultados esperados, por ello se plantea que la adopción de un ERP puede ser comprendida como un proceso co-evolutivo, ¿Pero por qué optar por este proceso? al parecer el patrón decisivo que dio como fracaso del ERP en esas empresas fue una mala implementación, mientras que en las otras empresas tuvieron una implementación exitosa gracias a la divulgación interna del uso del ERP, mostrando así, un enfoque acertado a la co-evolución. La co-evolución en un contexto empresarial son los cambios en las ideas y conocimientos de un empleado perteneciente a una organización a partir de la adquisición de conocimientos provenientes de otro empleado que integre la misma organización. Basándose en lo anterior sellos propusieron la capacitación en el uso del ERP de manera primordial a los empleados con mayores redes de afiliación formal e informal (los empleados con mayor número de subordinados así como los empleados con más vínculos afectivos dentro de la empresa) y controlar el flujo de información a sus compañeros de trabajo para que se inicie un proceso de propagación de información que fluya desde los empleados más influyentes hacia aquellos con los que interactúan. Para probar esta hipótesis ellos realizaron distintas simulaciones en Net Logo para simular dicho flujo de la forma descrita, resultando en que la información se propaga entre todas las personas al interior de la organización, similar a lo que ocurre con un virus o un rumor, (Rivera Silva, Vargas Reyes, y Bohórquez Arévalo, 2018). Aunque la propuesta se encuentra en fase conceptual y no se cuenta con investigación empírica que permita su validación es un hecho bastante interesante para tomarla en cuenta en este trabajo por su gran relevancia en la implementación real de este proyecto.

España *et al.*, presentan en "Patrón MVC, un componente para la implementación de una Estrategia Informática para mejorar gestión de datos en el área de estadística: Caso de Estudio Hospital Maternidad Babahoyo" los resultados de la investigación realizada en el Hospital Maternidad Babahoyo para desarrollar una estrategia informática que permita la gestión de datos de los pacientes, ya que todos sus registros se guardan en expedientes físicos y el consultarlos retrasa todas sus operaciones. Aplicaron investigación de campo, así como el desarrollo de encuestas y entrevistas a los miembros que brindan servicio en el hospital para saber qué información es la que se gestiona, coincidiendo que los registros de datos del paciente, producción de los médicos, informes de natalidad y mortalidad infantil, control de embarazo, partos, control de recién nacidos, consultas médicas y prescripciones médicas son parte fundamental para sus labores. Tras el análisis de los datos recopilados decidieron usar el patrón MVC para desarrollar su programa que llamaron "Software de Gestión de Clínicas (SIGEC)" ya que permite organizar los componentes aplicando ciertas normas de diseño y prepararlo para su evolución, además de terminar siendo de uso sencillo para usuarios finales, (España León, Gonzáles Valero, Mejía Viter, Campi Mayorga, y Campi Mayorga, 2016). Se integra la lectura de este artículo ya que muestra la gran utilidad de este patrón de arquitectura en proyectos especializados de gran magnitud.

Nieto en su artículo "Arquitectura por componentes JEE, un caso práctico" presenta los resultados de la implementación de un prototipo de software para gestionar comisiones de estudio y permisos académicos para los docentes de la Universidad Distrital Francisco José de Caldas. El proyecto tuvo por objetivo mostrar cómo integrar frameworks como Seam, EJB, Hibernate y RichFaces con la arquitectura por componentes JEE asi como describir sus principios y promover la reutilización, escalabilidad y mantenimiento de una aplicación de software mediante el ensamblaje de piezas de software reutilizables y facilitar el proceso de desarrollo, (Nieto Lemus, 2015). Se cita este articulo ya que los mismos principios de componentes se usa en este proyecto.

Nurzhan *et al.*, en "Comparison of JSON and XML Data Interchange Formats: A Case Study" realizan una comparativa entre JSON y XML en un entorno operativo usando como sujeto de estudio un programa cliente de red simple para transmitir objetos Java codificados en ambos formatos a un servidor. Para medir el desempeño de ambos se eligieron las siguientes métricas: número de objetos enviados, tiempo total para enviar el número de objetos, tiempo promedio por transmisión de objeto, utilización de la CPU del usuario, utilización de la CPU del sistema, utilización de la memoria. El primer escenario fue un cliente que envía un millón de objetos a un servidor utilizando tanto la codificación JSON como la codificación XML. El segundo escenario consistió en ejecutar una serie de casos de prueba con un número cada vez mayor de objetos, envía al servidor 20.000, 40.000, 60.000, 80.000 y 100.000 objetos codificados. Las mediciones de tiempo promedio y tiempo total proporcionan una indicación de que la velocidad de JSON supera la velocidad de XML. Además, JSON utiliza más recursos de CPU de usuario que XML en el escenario 1. La memoria depende del estado de los sistemas antes de un escenario o ejecución de prueba; sin embargo, el uso es similar entre JSON y XML. Los resultados indican que JSON es más rápido y utiliza menos recursos que su contraparte XML, (Nurseitov, Paulson, Reynolds, y Izurieta, 2009).

Martínez *et al.*, plantean en su artículo titulado "Diseño de Framework Web para el Desarrollo Dinámico de Aplicaciones" el diseño y creación de un Framework con base en herramientas de software libre para facilitar la adquisición de sistemas como este por empresas pequeñas, su estructura está bien definida y reusable permitiendo que sus componentes faciliten la creación de aplicaciones web, provee una capa de abstracción sobre la arquitectura original, ocultándola o adaptándola para no tener que utilizar el protocolo http de manera nativa y así acelerar los tiempos de desarrollo y mantenimiento de software. El Framework Web dispone de las siguientes funcionalidades: Proceso de autenticación, administración de roles y de usuarios, creación y administración de formularios, generación de formularios a partir de tablas, manejo de listas de valores y gestión de registros. Para complementar la arquitectura cliente/servidor y enfrentar diferentes problemas de interoperabilidad, seguridad, facilidad de acceso y desempeño, se definieron los siguientes componentes: Controlador, gestor de Seguridad, gestor de formularios, gestor de acceso a datos. Fue evidente la importancia de definir estándares de codificación y de administración del código fuente para lograr el nivel de calidad requerido en la implementación del Framework, (Martínez Villalobos,

Camacho Sánchez, y Biancha Gutiérrez, 2010). Se agrega este artículo porque muestra el interés de muchos desarrolladores y Organizaciones por la creación de contenido original y en base a herramientas de uso no comercial para su creación.

Ankurkar y Sable explican en "Evolving Ajax with JSON for Web Application Enrichment" los beneficios de utilizar las tecnologías AJAX y JSON juntas para dotar de un mejor rendimiento a las aplicaciones web en comparación de las otras aplicaciones tradicionales, también explican que el uso de AJAX común (con el formato de XML) tiene resultados menos eficientes que si se implementara JSON. Para demostrar su afirmación desarrollaron una aplicación y la compararon con otras aplicaciones web de servicios del gobierno y privados privadas mostrando la superioridad de la aplicación con diferentes extensiones, como la velocidad de la página (10 % más rápido que las otras), el número de solicitudes requeridas para recuperar la página, el tiempo de carga (200 % más rápida) y el tamaño de la página al reducir la cantidad de archivos, (Ankurkar y Sable, 2016).

Petković propone en su artículo "JSON Integration in Relational Database Systems" las ventajas de JSON en el uso dentro de las Bases de datos relacionales y en qué punto se ha hecho tan importante para los desarrolladores de software y la forma de cómo se aprovechan dichos beneficios en muchos gestores de bases de datos, también brinda ejemplos de las funciones especiales que se pueden realizar en Oracle con JSON que van desde las consultas de todos los datos en la cadena JSON hasta obtener un valor dentro de un campo especifico en él, (Petković, 2017). Esta lectura se elegido porque realiza algo parecido en la elaboración del presente proyecto.

Tabarés escribío "El surgimiento de HTML5; un nuevo paradigma en los estándares Web" en donde hace una retrospección en la historia del HTML5 y expone los detalles más sobresalientes de esta herramienta y su origen como una estandarización sucesora del HTML convencional. También plantea que antes de HTML5 estaba en desarrollo XHTML y que este no tuvo éxito, pero tras su aparición HTML5 tuvo un gran impulso gracias a sus características nuevas en combinación con CSS3 y JavaScript marcando una gran aportación social y creciendo con el respaldo de la comunidad desarrolladora y comercial, (Tabarés Gutiérrez, 2016).

Alonso en su documento "Responsive Web Design: Interfaces Web Adaptables al dispositivo empleando HTML5 y CSS3" muestra los resultados de su investigación de cómo conseguir que una página web se adapten a cualquier tipo de dispositivo, él indica que gracias a la evolución de HTML5 se puede lograr un diseño responsivo manipulando las propiedades de la página web usando las herramientas de CSS3 y explica cada una de ellas de forma detallada para todos los posibles escenarios de uso, (Alonso Vega, 2013).

Leprohon en "ECMAScript 6 and the evolution of JavaScript: A deeper look into the language's new features" analiza las nuevas características del lenguaje JavaScript desde sus

comienzos como un lenguaje meramente funcional hasta llegar a ser un lenguaje de programación con principios orientados a objetos y explica cómo afectan su estructura, propósito y naturaleza, al igual que demuestra las diferencias a través de código de todas las versiones de JavaScript hasta ECMAScript6 destacando la definición de clases, herencia, modularidad y operadores, (Leprohon Marc, 2017).

Capítulo 4

Metodología.

Por la propia naturaleza del paradigma POO empleado en este proyecto se ha estado trabajando bajo el análisis y diseño en espiral (Pressman, 2010), esencialmente por la capacidad de adaptación gradual de este enfoque de desarrollo, contrastado con el modelo en cascada se gana en flexibilidad, interacción y un mayor ajuste a las necesidades del cliente que pudieran surgir en el desarrollo del proyecto (Véase Figura 4.1). Como patrón de diseño se tomó como referencia el Modelo Vista Controlador (Fernández, 2012) para generar uno propio, que se adecue a las necesidades del proyecto. Se establecieron las siguientes premisas:

- Mediante la reutilización y el uso de la herencia reducir al mínimo los códigos generadores de la interfaz de usuario y también la transferencia de la información entre los clientes y el servidor.
- Un cliente web debe estar basado o extender un componente plantilla adaptativo.
- Los principios de la POO deben prevalecer a lo largo de todo el desarrollo con la finalidad de promover la creación de componentes de software y la reutilización.
- Todas las herramientas de desarrollo han de ser OpenSource en categoría de estables.
- La comunicación entre los clientes y el servidor siempre será asíncrona y solo será posible mediante el listener del módulo BackEnd, el cual árbitra el acceso al servidor.

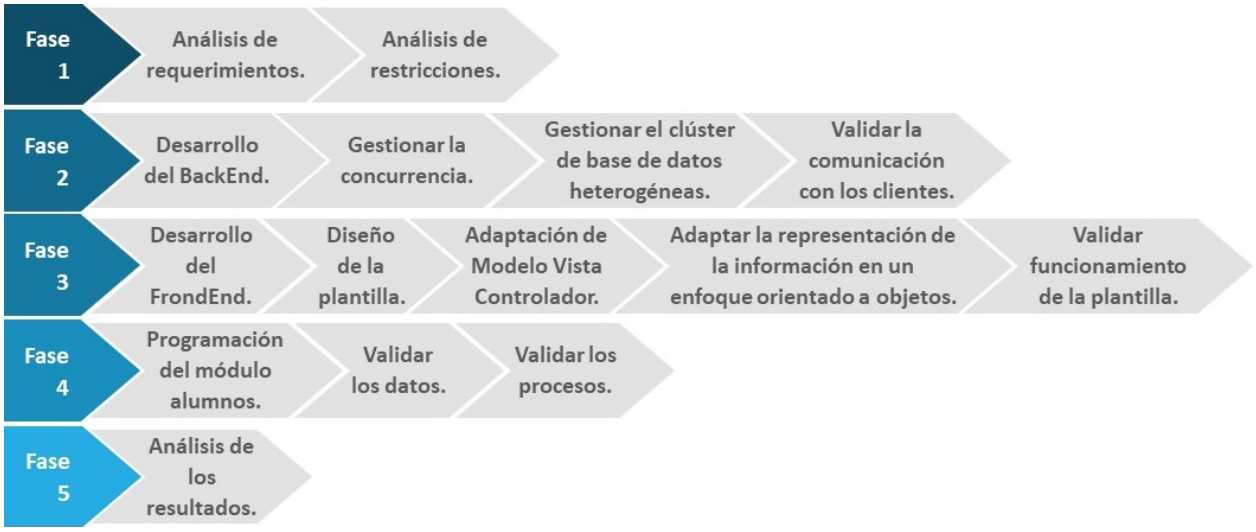

Figura 4.1: Diseño gráfico de la metodología.

Capítulo 5

Desarrollo de la solución.

5.1. Diseño del BackEnd.

Dada su magnitud, el BackEnd, fue enfocado de lo más simple a lo más complejo con la finalidad de poder solucionar el problema con un enfoque orientado a objetos y poder reutilizar los componentes siempre. Secuencialmente fueron programadas las siguientes clases en PHP quedando de forma abierta para las adecuaciones necesarias que mandara el proceso de diseño en espiral:

Campo: Constituye el molde para crear la unidad mínima que puede ser representada en un sistema de bases de datos, el campo. Su constructor permite definir el nombre, el tipo de dato e indicar si se trata de un campo clave. Posee también un método con el cual se puede generar una representación JSON (Ecma, 2017) de sí mismo.

Registro: Encapsula el conjunto de campos que definen un registro de base de datos. Emulando polimorfismo, su constructor permite crear objetos de diferentes maneras para satisfacer las necesidades de construcción que fueron detectadas durante el diseño. Posee también un método para crear un objeto JSON con la descripción del registro y los valores de sus campos. Estos objetos también sirven de contenedores para las consultas o procedimientos almacenados donde el resultado es un valor o un solo registro. Adicionalmente es capaz de generar el código SQL de las consultas SELECT, INSERT, UPDATE y DELETE tomando como base su propia estructura y los valores de sus campos.

Tabla: Los productos de esta clase son capaces de almacenar tantos objetos de tipo Registro como sean necesarios para describir la estructura de una Tabla de base de datos, pueden ser empleados para ejecutar procedimientos almacenados de la base de datos a la cual hayan sido asociados en donde el resultado sea un valor, un registro o un conjunto de registros y son capaces de contener el dataset resultante de una consulta SQL en un array dinámico de objetos de tipo Registro.

BD: Esta clase es una extensión de la librería PDO (Popel D., 2007) de PHP 7.0 que consigue conectar de forma transparente a diversos gestores de bases de datos. Es capaz de contener un array de objetos Tablas o resultados de consultas que pueden ser cargadas de inicio para poderlas enviar al cliente como un paquete de datos de tipo JSON. Puede ejecutar las operaciones SELECT, INSERT, UPDATE Y DELETE de un registro ó una o más tablas.

5.1.1. Gestión de la concurrencia.

En un gran sistema, como es el caso del SITM, la gestión de la concurrencia a las bases de datos es un factor determinante para su correcto funcionamiento. En el SITM la gestión de la concurrencia se lleva a cabo mediante la técnica o patrón conocida como singleton, la cual nos permite gestionar las conexiones para hacer posible que exista una y solo una conexión por cliente, independientemente del número de solicitudes que éste realice. Las clases descendientes de la clase BD del proyecto quedan habilitadas para implementar un singleton y así evitar la sobrecarga de conexiones que podría ocasionar una solicitud masiva de peticiones de un cliente. Por defecto, el constructor de la clase conecta con un gestor MySQL y a petición se puede realizar la conexión a otros gestores como son MSSQL, Oracle, PostgreSQL, SQLite, etc., de forma transparente. Las siguientes líneas muestran el singleton creado para conectar con un gestor de base de datos MSSQL ubicado en un segundo servidor. Todos los singleton pertenecen a clases derivadas de la clase BD en PHP.

```
1 private static $BD;
2 public static function BD ){
3   if (!isset(self::$BD))
4     self::$BD = new BD("E00758\SMISANTLA","ga","use","clave","MSSQL");
5   return self::$BD;
6 }
```

Índice de algoritmos 5.1: Singleton para el manejo de la conexion a MSSQL.

5.1.2. Gestión del clúster de base de datos heterogéneas y comunicación con los clientes.

Se tomó la decisión de crear un cluster de base de datos y no una mega base de datos para simplificar su gestión, mantenimiento y primordialmente mantener la capacidad de interactuar con otros sistemas sin la necesidad de llevar a cabo una migración de datos o adecuar las consultas a cada gestor, simplemente solicitando la ejecución de procedimientos almacenados alojados en cada instancia, favoreciendo colateralmente un acceso seguro a los datos.

Se programó en PHP la clase "Proceso" con la finalidad de que sea extendida para crear clases que definan el comportamiento de objetos cuyas funciones serán trabajar como un listener de las peticiones POST que llegan al servidor y árbitro de acceso a las bases de datos. Las extensiones de esta clase, verifican que la solicitud sea una operación reconocida, que la naturaleza de los datos de solicitud sea la esperada y si todo es correcto, es él, y no el cliente, quien accede al cluster para entregar los resultados de la consulta solicitada. Si la solicitud no es reconocida, el listener asumirá una amenaza de ataque y devolverá un mensaje de error que quizás terminaría ocasionando un mal funcionamiento en el cliente. Cada cliente que se comunique con el server debe hacerlo mediante una petición asíncrona de tipo AJAX (Woychowsky, 2008) y enviar/recibir datos en formato JSON. La interacción del listener con el cluster de bases de datos y los clientes queda representada en la Figura 5.1.

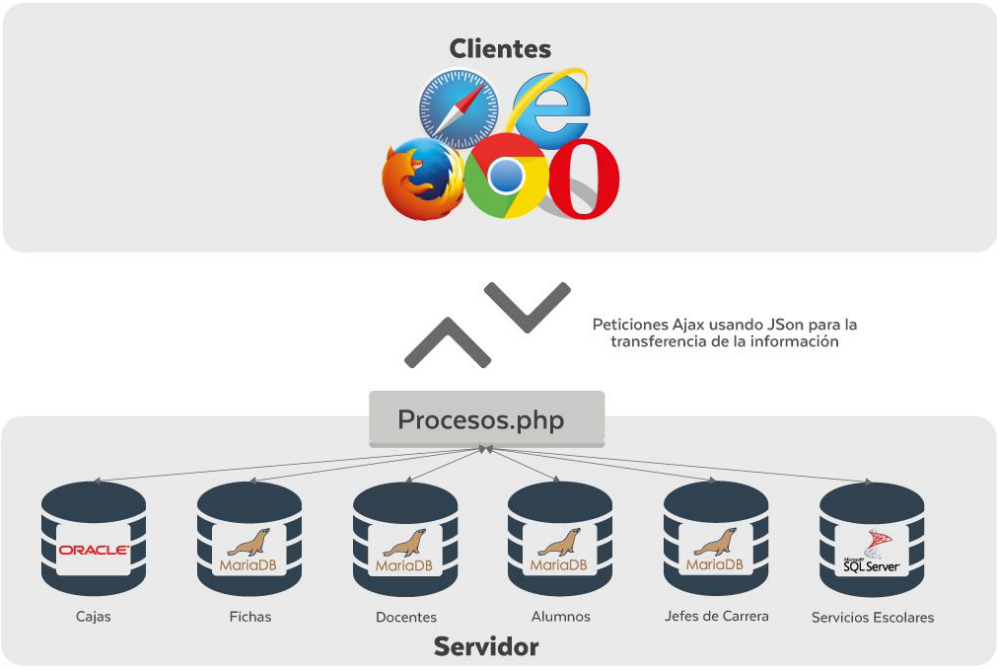

Figura 5.1: Interacción del BackEnd y el FrontEnd.

5.2. Diseño del FrontEnd.

Se tomó como punto de partida el patrón de arquitectura: Modelo Vista Controlador y se adaptó un motor de comunicaciones asíncronas propio basado en Ajax (Véase Figura 5.2).

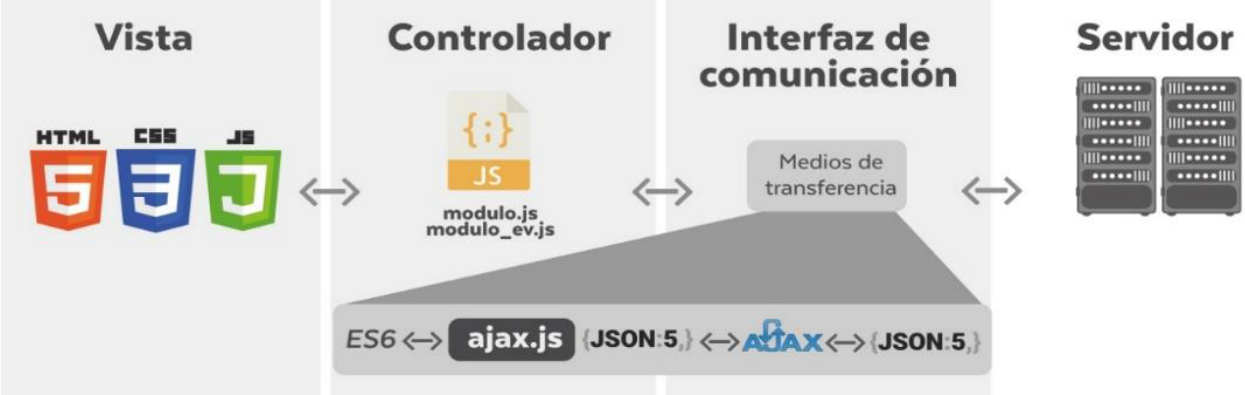

Figura 5.2: Adaptación del MVC al proyecto SITM.

5.2.1. Vista.

Los clientes fueron creados bajo los principios de la web adaptativa establecidos por Ethan Marcotte en 2009 (Marcotte, 2011). Marcotte indica que existen tres elementos fundamentales para lograr un diseño web adaptativo, que son: cuadrícula fluida, imágenes flexibles y media queries. Gran parte de la solución la proporciona un buen manejo de CSS3 Flexbox y una parte la programación en Javascript para adaptar la interfaz de usuario a las resoluciones de pantalla que tienen los diferentes dispositivos (Véase Figura 5.3).

Figura 5.3: Diseño Web Responsivo.

Representación de la información orientada a objetos.

Dado el número de veces que se ocupa la terna campo, registro, tabla (Véase Figura 5.4), se programaron las tres clases siguientes en JavaScript con el fin de conseguir un modelo orientado a objetos de la información a representar:

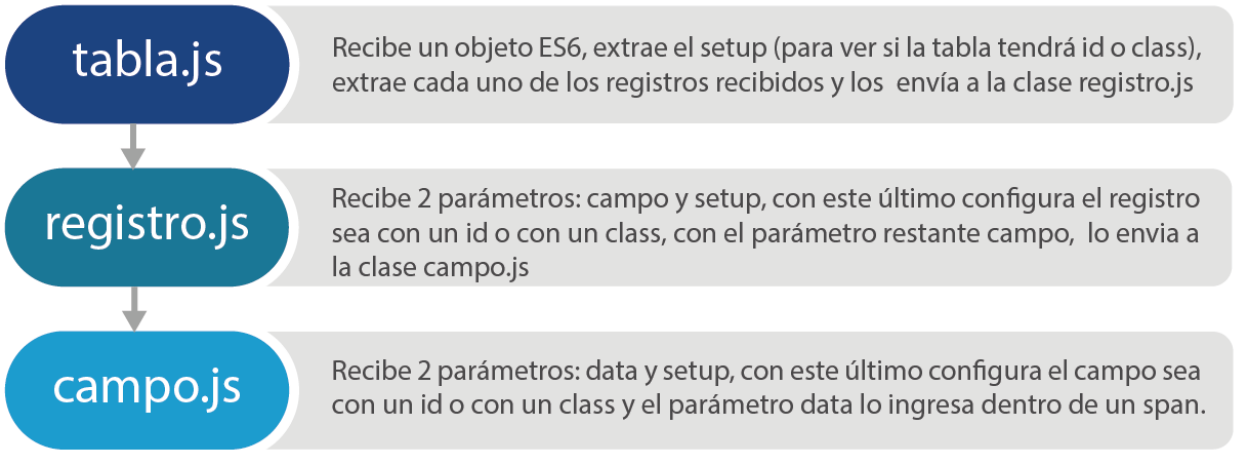

Figura 5.4: Representación Orientada a Objetos de la información (Tabla, Registro y Campo).

Campo: Esta clase se utiliza para crear un campo que es la unidad mínima que contiene el dato JSON que estamos recibiendo, en el constructor se recibe nombre y contenido de dicho campo, estos datos se muestran en la interfaz por medio de un contenedor de tipo span.

Registro: Permite la agrupación múltiples objetos de tipo Campo, tantos como sea necesario incluir dentro de un mismo registro, para mostrarlos en la interfaz se utiliza un contenedor de tipo div.

Tabla: Con esta clase se define una estructura que puede agrupar uno o más objetos de tipo registro según se requiera, esta agrupación que se hace con la ayuda del contenedor div permite representar la información al usuario final como una tabla. Un ejemplo de aplicación de este conjunto de clases lo representa la forma en que se gestionó la descripción de las materias y su conjunto, como se muestra en la Figura 5.5.

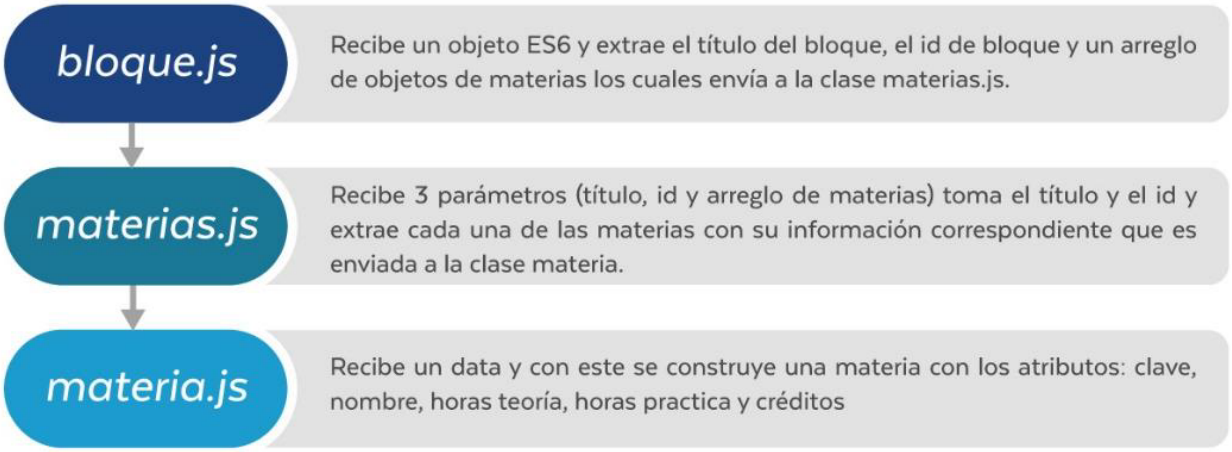

Figura 5.5: Representación Orientada a Objetos de la información (bloque, materias, materia).

5.2.2. Controlador.

El control de la vista se lleva a cabo mediante la instanciación de tres clases escritas en JavaScript cuya finalidad es la siguiente:

modulo_ev: Es la clase que se encarga de la gestión de los eventos generados por el usuario.

modulo: Es una extensión de la clase sitm.js, que lleva a cabo la actualización de la vista y mantiene las callbacks resultantes de las solicitudes Ajax dirigidas al servidor.

sitm: Es el componente por el cual, mediante su extensión, sirve de plantilla a la vista manejada por la instancia de la clase modulo.js. Es también la responsable de: designar la estructura y áreas correspondientes para insertar cada componente, definir la estructura de los componentes principales (Menú lateral, Menú superior y encabezado), definir el comportamiento y las propiedades de cada componente principal, realizar el insertado de cada componente principal al finalizar su creación (Véase Figura 6.1 y Anexos 8.8, 8.9 y 8.10).

Capítulo 6

Resultados.

6.0.1. Interfaz de Comunicación.

El segmento de interfaz de comunicación está constituido por la clase denominada ajax.js que crea y envía peticiones asíncronas de métodos AJAX consumiendo objetos tipo EcmaScript6 que son convertidos a JSON para ser enviados al servidor, este a su vez devuelve una respuesta en el mismo formato que convierte en objetos EcmaScript6 para ser dirigidos al cliente (Véase Figura 6.1).

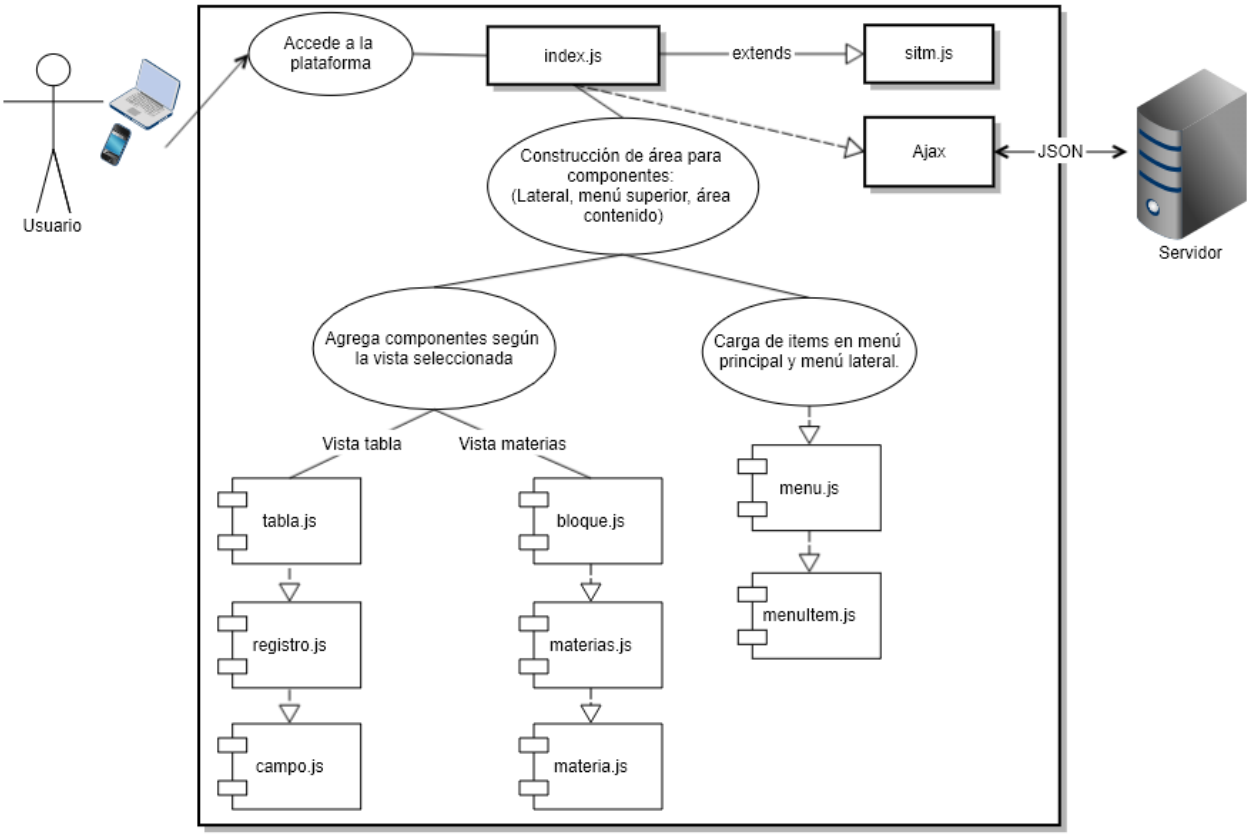

Figura 6.1: Diagrama descriptivo de la plantilla.

La Figura 6.1 muestra como el usuario entra con un dispositivo al sistema a través de internet, el navegador web carga el archivo index, el cual hereda de la plantilla "sitm.js" funciones dentro ella, posteriormente obtiene la información en la base de datos por Ajax. Al tener toda la información completa empieza a construir la estructura HTML de la plantilla y agrega los componentes en el área de contenido designada.

6.0.2. Funciones en el módulo.

El proceso más importante dentro del módulo del alumno es la inscripción online, para ello es necesario el proceso mostrado en la figura 6.3 comenzando por el catálogo de las materias para seleccionar (Figura 6.2), este se conforma por los cursos disponibles y se muestra de forma ordenada según el kardex del alumno, con él se puede agregar a la carga académica del alumno todas las materias disponibles a excepción de las materias obligatorias, estas están seleccionadas desde el principio y no se pueden desmarcar. Una vez seleccionadas las materias haciendo un total de créditos válido, el alumno puede seguir con el proceso de inscripción solo si no se detecta algún choque de horario entre ellas, si existe un choque no puede continuar con este proceso, a no ser que cuente con un permiso para él proporcionado por el jefe de carrera. Una vez comprobado que no hay problemas (No hay choque de horarios o cuenta con el permiso del jefe de carrera) el alumno puede terminar con su inscripción.

Figura 6.2: Materias disponibles.

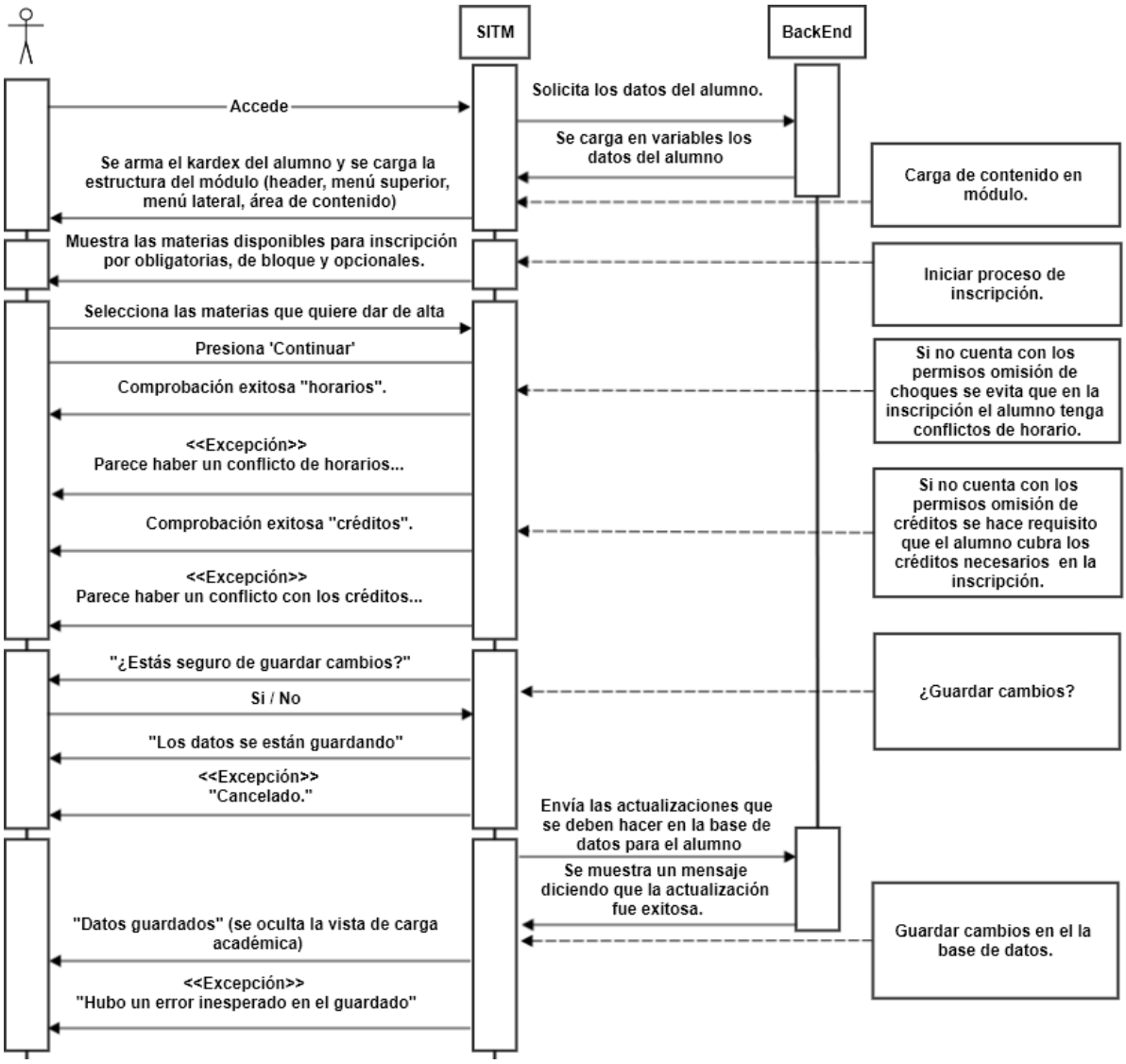

Figura 6.3: Diagrama de secuencia de la inscripción del alumno.

En la Figura 6.3 se representa el proceso de inscripción del alumno a través del sitm. Cuando el alumno inicia este proceso se carga su información y los datos del kardex (Véase las Figuras 6.7, 6.8 y 6.9) base a ellos se muestran todas las materias que el alumno puede escoger para el semestre que debe cursar, una vez seleccionadas las materias, el alumno debe presionar el botón continuar. Si no existen problemas con los horarios o créditos necesarios el alumno se podrá inscribir, de lo contrario aparecerá un error y el proceso se detendrá aquí (el alumno puede solicitar un permiso para el sistema con el jefe de carrera y evitar que siga bloqueando el proceso de inscripción en esta parte). Llegado a este punto solo se necesita recibir la autorización del usuario para guardar los cambios, si es así y no ocurre ningún error de comunicación con el servidor los datos de la base de datos se actualizan y se cierra la ventana de inscripciones, pero si no es permitido este proceso solo será cancelado.

También las materias seleccionadas se visualizan en la parte inferior en forma de horario en donde se muestra si existen choques de horario o no (Figura 6.4).

Figura 6.4: Vista previa de horario.

En la sección de "Servicios Escolares" se encuentra la vista "Kardex" (Figura 6.5) con ella se puede ver todas las materias de forma ordenada y la opción de filtrarla por materias cursadas, sin cursar, aprobadas y sin reprobar para mayor comodidad del alumno.

Figura 6.5: Vista Kardex.

En la sección con el nombre “Institucional” se puede ver los adeudos del alumno, estos se dividen en: evaluaciones o materiales (Figura 6.6).

Figura 6.6: Vista adeudos.

La información del kardex del alumno es un conjunto de datos que se muestran en forma de oficio, estos datos no se encuentran en una sola tabla dentro de la base de datos, por ello es necesario juntarlos para ser usados de una forma estructurada y ordenada, mucha información puede ser extraída directamente de la base de datos mientras que otra debe ser derivada (como los créditos cursados, los creditos sin cursar, etc...). La clase kardex se encarga de esta tarea y puede ser utilizada en otros módulos si se necesita disponer de ella en un futuro (Véase las Figuras 6.7, 6.8 y 6.9), esta clase se explica a continuación:

Primero se definen las variables:

```
let materias = kardex.ordenarMaterias()[0],
  reticula = gd.html_getItem("materias"),
  plan = gd.html_getItem("perfil").carreras[0],
  personales = gd.html_getItem("personales")[0],
  carreras = gd.html_getItem("carreras"),
```

- “materias” es el contenedor del historial ordenado cronológicamente de todas las materias del alumno (una materia puede estar en blanco simplemente por no tener ningún registro, de lo contrario tendría el indicador de si está o no siendo cursada, las oportunidades que curso para aprobar, la calificación o acreditación de cada una de esas

oportunidades y en qué periodo se tomó cada una de ellas) además de tener el resultado de la cantidad de materias sin considerar en el Kardex.

- "reticula" son los datos de cada materia en la carrera.
- "plan" es el nombre código del plan académico de la carrera.
- "personales" son los datos personales del alumno.
- "carreras" es la información de las carreras en el tecnológico.

Las siguientes variables tendrán designado un valor más adelante y solo son definidas al principio.

```
1  pctje, carrera, cal ,
```

- "pctje" es el porcentaje de créditos aprobados por el alumno.
- "carrera" es el nombre de la carrera del alumno.
- "cal" es la calificación de cada materia.

Después siguen las variables que serán calculadas, es decir que comienzan con un valor de 0.

```
1  creTo = 0 , creAc = 0 , creCu = 0 , totalMat = 0 ,
2  sumaCalSR = 0 , sumaCalCR = 0 , promSinR = 0 , promConR = 0 ,
3  acreditadas = 0 , cursadas = 0 , aprobadas = 0 ,
```

- "creTo" son la cantidad de créditos en toda la carrera que debe cubrir el alumno (los créditos son cubiertos al aprobar la materia a la que correspondan).
- "creAc" son la cantidad de créditos cubiertos en materia acreditables (materias que no tienen una calificación solo se marcan como acreditadas) por el alumno.
- "creCu" son la cantidad de créditos cursados por el alumno.
- "totalMat" es la cantidad total de materias del alumno.
- "sumaCalSR" es la suma de las calificaciones de las materias aprobadas por el alumno.
- "sumaCalCR" es la suma de todas las calificaciones del alumno.
- "promSinR" es el promedio de las calificaciones de las materias aprobadas por el alumno.
- "promConR" es el promedio de todas las calificaciones del alumno.
- "acreditadas" es la cantidad de las materias acreditadas por el alumno.

- “cursadas” es la cantidad de las materias cursadas por el alumno.
- “aprobadas” es la cantidad de las materias acreditadas por el alumno.

```
1   sinConsiderar = materias.sinConsiderar;
```

Finalmente se define la variable “sinConsiderar” con el valor extraído de la cantidad de materias sin considerar en la variable “materias”.

Después se inicia con las operaciones:

```
1 delete materias.sinConsiderar;
```

Se elimina la cantidad de materias sin cursar en la variable “materias” pues ya no es de utilidad.

```
1  for(let materia in materias){
2    if(materias[materia]!="-")
3      totalMat++;
4    if(materias[materia].CURSOS != undefined){
5      let op1=materias[materia].CURSOS[0],
6        op2=materias[materia].CURSOS[1],
7        op3=materias[materia].CURSOS[2];
8      if(op1.CA == "ACR" || op2.CA == "ACR" || op3.CA == "ACR")
9        cal = "ACR";
10     else
11       cal = Math.max(op1.CA , op2.CA , op3.CA);
12     if(cal != -1 && cal != "ACR")
13       sumaCalCR = sumaCalCR + cal;
14     if(cal >= 70 || cal == "ACR"){
15       aprobadas++;
16       if(cal != "ACR")
17         sumaCalSR = sumaCalSR + cal;
18       else
19         acreditadas++;
20     }
21     if(op1.PE != "")
22     cursadas++;
23   }
24   for(let cd in reticula){
25     if(reticula[cd].clave == materia){
26       creTo = creTo + reticula[cd].creditos;
27     if(cal >=70 || cal == "ACR")
28       creAc = creAc + reticula[cd].creditos;
29     if(materias[materia].CU == 1)
30       creCu = creCu + reticula[cd].creditos;
31     }
32   }
33 }
```

Se recorre cada uno de los elementos en la variable “materias”, cada uno de estos elementos es una materia, entonces se procede a lo siguiente por cada una:

Si el alumno debe tomar la materia, “totalMat” aumenta en 1 su valor (para saber si el alumno debe cursar o no la materia esta debe tener un valor diferente a ‘-’, esto es debido a que esas materias no pertenecen a su especialidad).

Si la materia fue cursada entonces se procede a extraer su calificación y almacenarla en la variable “cal”, después se evalúa de forma condicional esta variable:

Si “cal” tiene un valor de evaluación entre 0 y 100 entonces “sumaCalCR” incrementa de valor sumando la cantidad de “cal”.

Si “cal” tiene una cantidad mayor de 70 o “cal” tiene el valor ‘ACR’ (acreditado) entonces la variable “aprobadas” aumenta su valor sumando 1, pero si “cal” es diferente de ‘ACR’ entonces “sumaCalSR” aumenta su valor sumando el valor de “cal” de lo contrario “acreditadas” aumenta su valor sumando 1.

Si la materia fue cursada entonces “cursadas” aumenta su valor sumando 1. Se recorre cada uno de los elementos en la variable “reticula”, cada uno de estos elementos es una materia de la carrera con toda su información básica, entonces se procede a lo siguiente por cada una:

Primero se comprueba que la materia en “reticula” sea la misma materia de “materias” que se está recorriendo esto se hace a través de su clave, cuando es así “creTo” aumenta su valor sumando la cantidad de créditos correspondiente a la materia (los créditos solo pueden ser consultados de “reticula”). Nuevamente se entra a una evaluación condicional con “cal”, Si “cal” tiene una cantidad mayor de 70 o “cal” tiene el valor ‘ACR’ entonces “creAc” aumenta su valor sumando la cantidad de créditos correspondiente a la materia. Si la materia está siendo cursada entonces “creCu” aumenta su valor sumando la cantidad de créditos correspondiente a la materia.

```
1 promSinR = sumaCalSR / (aprobadas-acreditadas);
2 promSinR = promSinR.toFixed(2);
3 promConR = sumaCalCR / (cursadas-acreditadas);
4 promConR = promConR.toFixed(2);
5 pctje = creAc*100/creTo;
6 pctje = pctje.toFixed(0);
```

Con las variables obtenidas anteriormente se calculan las variables “promSinR”, “promConR”, “pctje”.

```
1 for(let i in carreras){
2   if(carreras[i].plan == plan)
3     carrera = carreras[i].descripcion;
4 }
```

En esta parte del código solo se consigue el nombre de la carrera usando el nombre del plan académico, el resultado es almacenado en la variable “carrera”.

```
let size = Object.keys(materias).length - 1,
  inicio = Object.keys(materias)[sinConsiderar],
  fin = Object.keys(materias)[size],
```

Se definen las variables “inicio” así como “fin”.

```
  p1pm = materias[inicio].CURSOS[0].PE,
  p2pm = materias[inicio].CURSOS[1].PE,
  p3pm = materias[inicio].CURSOS[2].PE;
if(p3pm != "")
  inicio = p3pm;
else if (p2pm != "")
  inicio = p2pm;
else
  inicio = p1pm;
if(materias[fin].CURSOS != undefined){
  let op1um = materias[fin].CURSOS[0],
    op2um = materias[fin].CURSOS[1],
    op3um = materias[fin].CURSOS[2];
  if(op1um.CA == "ACR" || op2um.CA == "ACR" || op3um.CA == "ACR")
    cal = "ACR";
  else
    cal = Math.max(op1um.CA , op2um.CA , op3um.CA)
  if(cal >=70 || cal =="ACR"){
    if(op3um.PE != "")
      fin = op3um.PE;
    else if (op2um.PE != "")
      fin = op2um.PE;
    else
      fin = op1um.PE
  }
  else fin = "";
}
```

La variable “inicio” tendrá como valor el periodo en el cual el alumno inicio su carrera. La variable “final” tendrá como valor el periodo en el cual el alumno egreso de su carrera.

```
let data = { nControl: personales.login , nombre : {nm : personales.nombre.nm
    , pa : personales.nombre.pa , sa : personales.nombre.sa} , carrera :
    carrera , sem : personales.semestre , sit: personales.situacion, plan :
    plan , ingr : inicio, term : fin , totalMat : totalMat , acreditadas :
    acreditadas , aprobadas : aprobadas , cursadas : cursadas , pctje :pctje ,
     creTo : creTo , creAc : creAc , creCu : creCu , promSinR : promSinR ,
    promConR : promConR , materias : materias};
return data;
```

Se une toda la información en un objeto llamado “data” que termina siendo retornado.

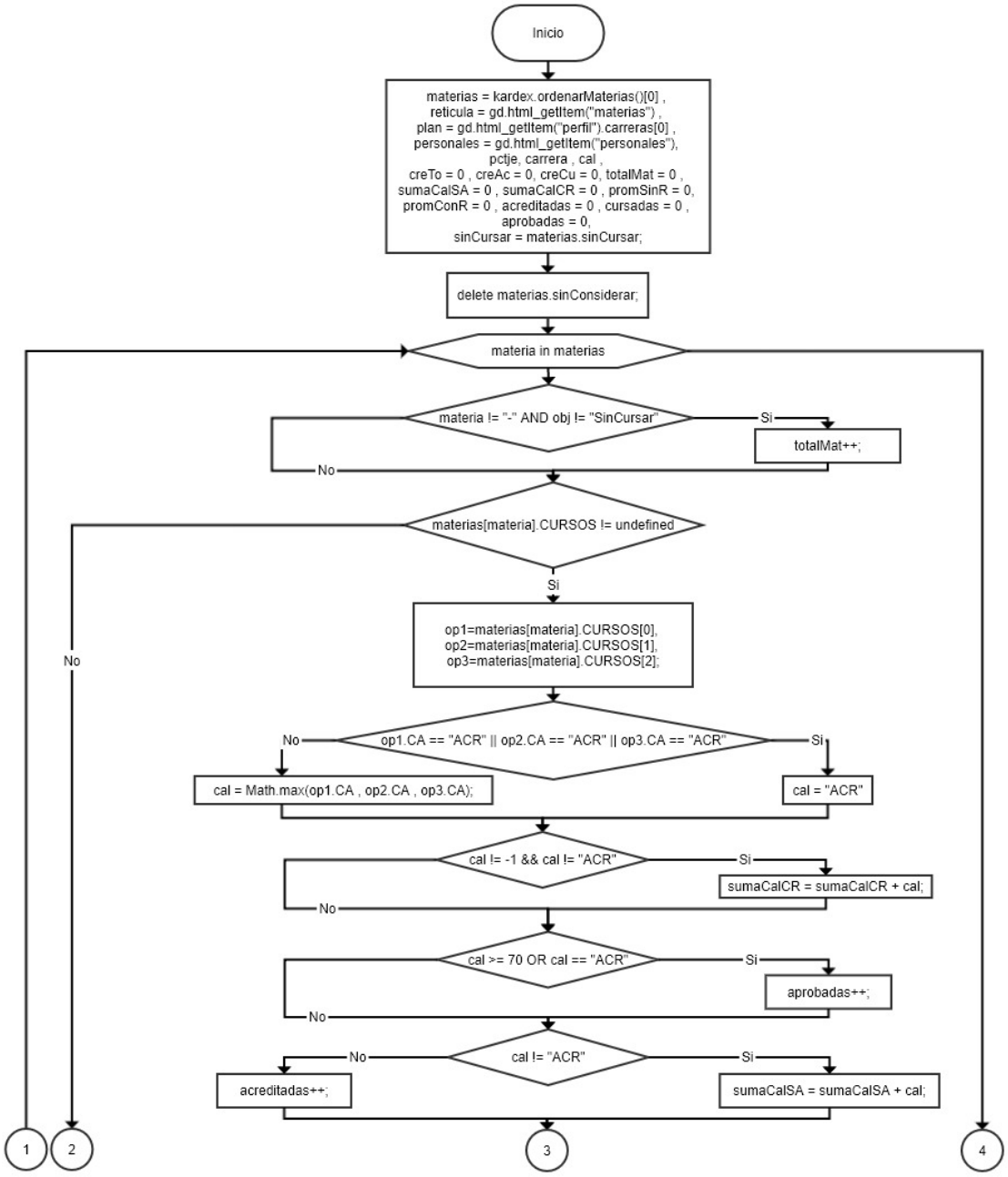

Figura 6.7: Diagrama de flujo para usar el kardex (1ra parte).

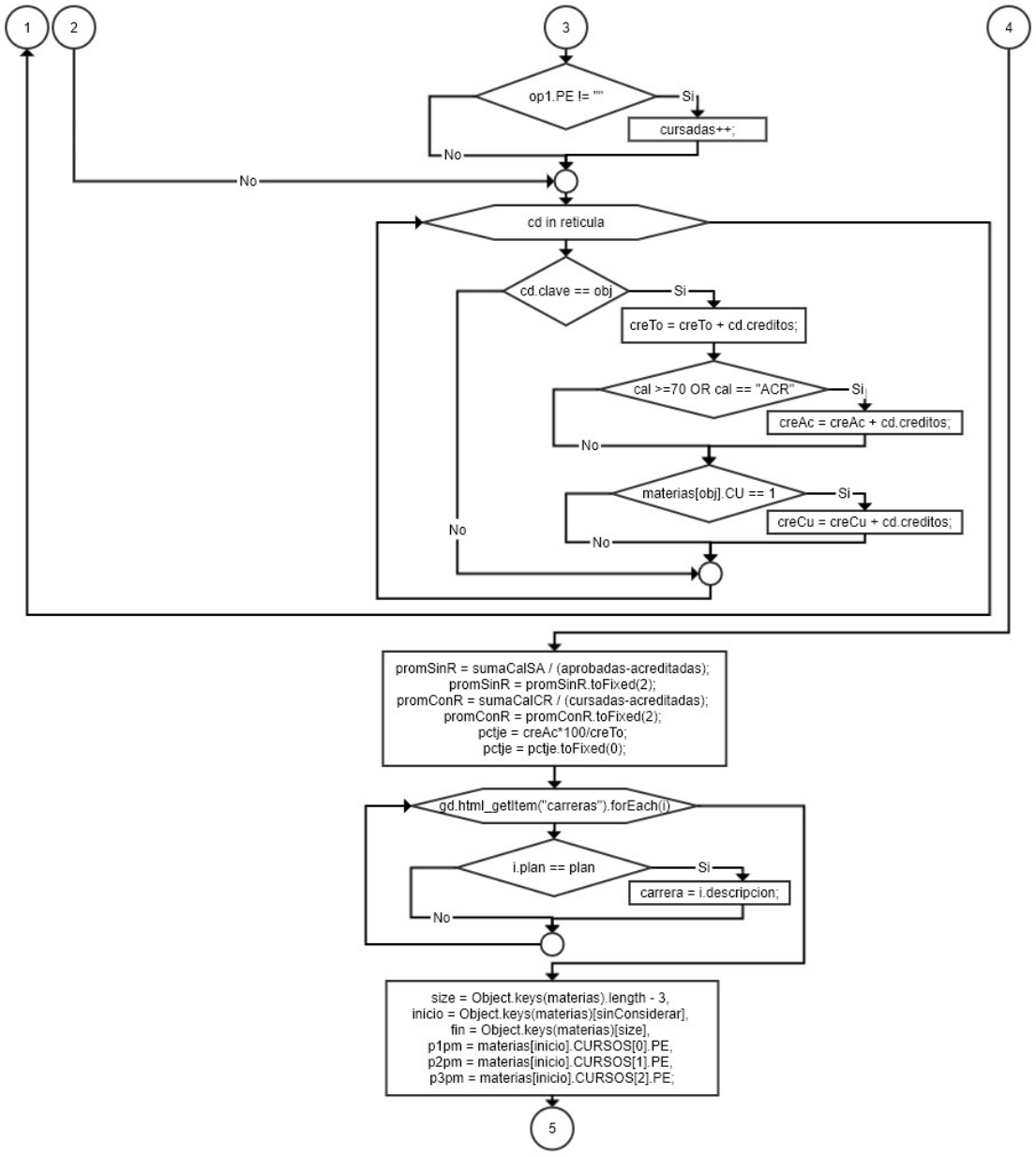

Figura 6.8: Diagrama de flujo para usar el kardex (2da parte).

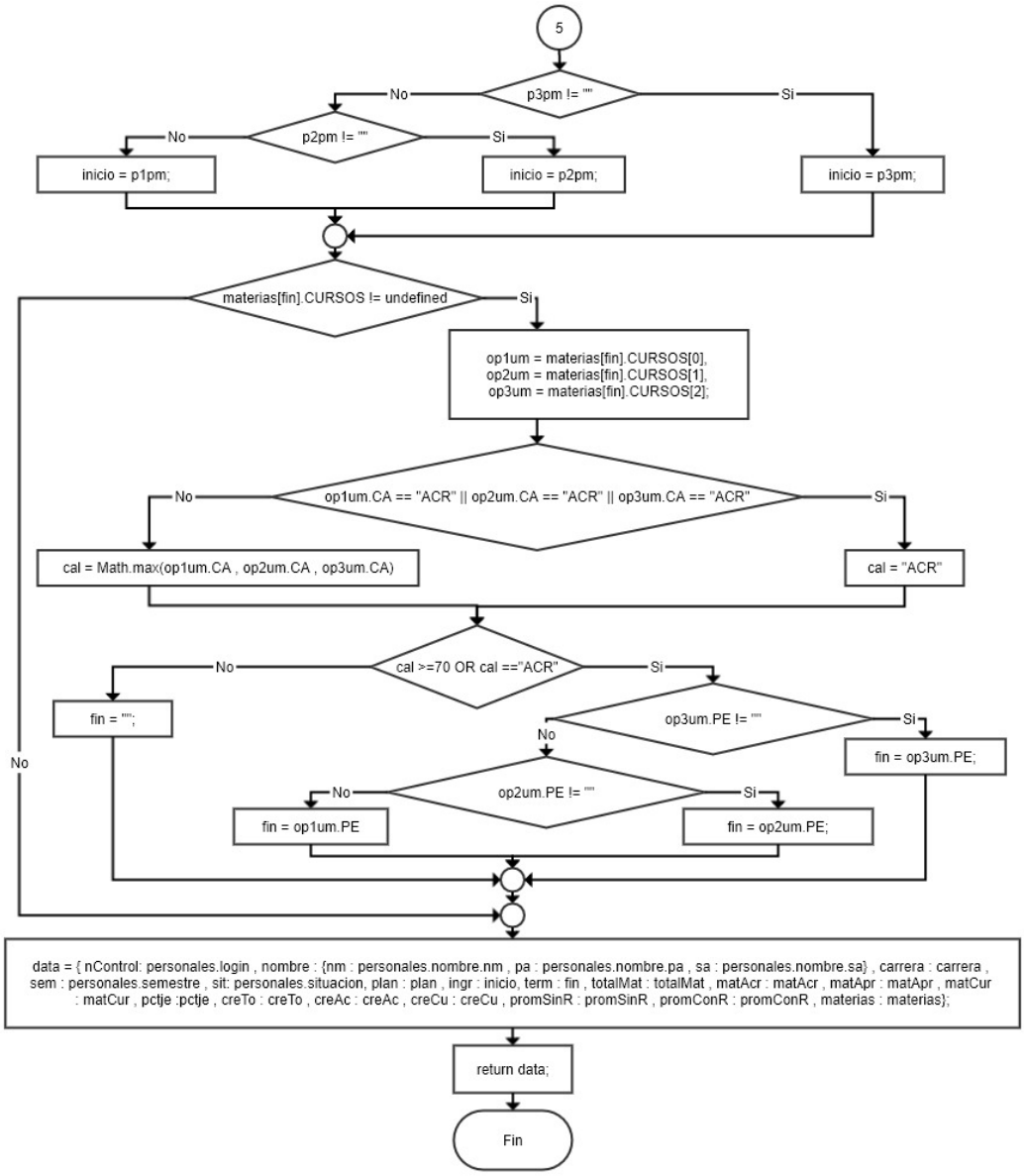

Figura 6.9: Diagrama de flujo para usar el kardex (3ra parte).

Capítulo 7

Análisis de resultados.

7.1. Comparación entre otras librerías y las librerías desarrolladas.

Al comenzar el proyecto se utilizaban diferentes librerías disponibles para uso libre de desarrollo web, sobre todo la librería **jQuery**, la cual era integrada en la mayoría del código dentro del sistema, pero tras un análisis deductivo se decidió extraer dicha librería y solo utilizar las funciones proporcionadas en EcmaScript6, ya que la dependencia de las librerías como jQuery causaba que la carga de los recursos fuera más tardada, haciendo esperar al cliente otros datos innecesarios para la utilización de pocas funciones. El ejemplo más sencillo de esto es una librería para generar cuadros de dialogo de distintos tipos en el sistema, la cual fue nombrada simplemente **"mensajes.js"**, antes de ella se utilizaba la librería **"alertify"** por ser la librería más popular para este tipo de proyecto, pero tras las razones ya mencionadas se sustituyó. También existen librerías como **Bootstrap-dialog** que forma parte del Framework Bootstrap que utiliza jQuery y esto causa un funcionamiento más lento además de aumentar la cantidad de datos necesitada durante la descarga. En la Tabla 7.2 se observa que "mensajes.js" es al menos 95 % más ligero y eficiente en carga que las otras librerias mencionadas por utilizar solamente EcmaScript6 o por ser de un menor tamaño.

	mensajes	alertify	Bootstrap-dialog
Versión	1	1.11.0	3
Líneas de código	102	3595	1394
Tamaño (kb)	8	136	47
Responsivo	Si	Si	Si
Uso JQuery	No	No	Si
Comparación de velocidad en 1000 ejecuciones			
Mínimo (ms)	0	0	3.9999999999054126
Máximo (ms)	8.000000000038199	24.00000000034197	68.000000000211
Promedio (ms)	2.561000000000000263	0.271999999999025	10.017000000000735

Cuadro 7.1: Tabla comparativa de librerias.

7.2. Análisis de Seguridad Web.

Acunetix Vulnerability Scanner [1] es un aplicación sobre un entorno web que realiza pruebas de seguridad de aplicaciones web automazadas. Se han sometido a un análisis las aplicaciones web de SII Minatitlan y SII Veracruz y Ciudad Madero. A continuación se comparan las vulnerabilidades de cada sistema SII y del SITM.

Sistema / Vulnerabilidades	Alta	Media	Baja	Informacional
SITM	0	0	3	0
SII Madero	5	3	3	1
SII Veracruz	0	6	2	4
SII Minatitlán	5	5	3	2

Cuadro 7.2: Tabla comparativa con otros sistemas implementados en otros tecnológicos.

Las alertas detectadas en el reporte de Acunetix son del tipo "Tiempo de respuesta lenta" e informa que "Este tipo de archivos se pueden orientar en ataques de denegación de servicio. Un atacante puede solicitar esta página repetidamente desde múltiples computadoras hasta que el servidor se sobrecargue" y correponden a los 3 siguientes ficheros.

- /alumnos/js/funcionesKardex.js
- /alumnos/js/horarios.js
- /alumnos/js/TablaKardex.js

Aunque es cierto que el tiempo de respuesta es mucho menos rápido que los otros ficheros es completamente comprensible, ya que estos se encargan de adaptar, filtrar o sencillamente son los archivos con más operaciones requeridas por el sistema, pero estos no trabajan del lado del servidor sino del lado del cliente, además se optimizó las operaciones en estos códigos para reducir el costo computacional así que el riesgo establecido es el mínimo.

[1] Software para auditar seguridad Web, disponible desde `https://www.acunetix.com/vulnerability-scanner/` versión trial

7.3. Comparación entre el estado del arte y el sistema ITSM.

Aunque no se cuenta con la suficiente información técnica de algunos de los sistemas publicados en los artículos dentro del estado del arte, aún es posible realizar una pequeña comparación cualitativa de esos resultados con los obtenidos en el desarrollo del SITM. A continuación se muestra la tabla que presenta dicha comparación.

	Omite el uso de librerías de terceros.	Es orientado a objetos.	Funciona bajo MVC.	Tiene diseño responsivo.
"SITM"	SI	SI	SI	SI
"Patrón MVC, un componente para la implementación de una Estrategia Informática para mejorar gestión de datos en el área de estadística: Caso de Estudio Hospital Maternidad Babahoyo", (España León, Gónzales Valero, Mejía Viter, Campi Mayorga, y Campi Mayorga, 2016).	NO	NO	SI	NO
"Arquitectura por componentes JEE, un caso práctico", (Nieto Lemus, 2015).	NO	SI	NO	NO
"Diseño de Framework Web para el Desarrollo Dinámico de aplicaciones", (Martínez Villalobos, Camacho Sánchez, y Biancha Gutiérrez, 2010).	SI	SI	NO	NO

Cuadro 7.3: Tabla comparativa con otros sistemas implementados en el estado del arte.

Capítulo 8

Conclusiones y trabajos futuros.

8.1. Conclusiones.

Como resultados se han obtenido: un modelo de comunicación con el servidor muy simple, transferencia de códigos con datos reducidos al mínimo, una eficiente renderización de los componentes de interfaz y de forma inherente se ha elevado la seguridad del sistema.

- El desarrollo de este proyecto es un punto de partida para lograr la automatización de los diferentes servicios que al sistema integral le competen, sumado a lograr la unificación de los diferentes departamentos solidificando un formato global para la consolidación de datos del alumno.
- El proyecto buscó adaptarse a las tecnologías emergentes, alternando entre los dispositivos que más frecuentan adaptándose a las necesidades del usuario por realizar transacciones de información más fácilmente e incluso en la palma de su mano, el uso de Flexbox en gran medida fue la solución para que la distribución de la página se adaptara al tamaño de la pantalla del dispositivo.
- Con respecto al BackEnd, este se encuentra terminado en su totalidad, superando diversas pruebas de funcionamiento y, además, está operando de manera adecuada en otros proyectos.
- Mediante la creación de componentes dinámicos, se logra reducir más del 90 % el código, esto comparado con implementaciones hechas en versiones anteriores, con esto se obtiene una menor taza de envío/recepción de datos agilizando la carga y acceso a los diferentes módulos.
- El componente principal del FrontEnd (plantilla) opera ya satisfactoriamente, se mantiene sujeto a las adecuaciones que requiera el desarrollo en espiral.
- La sustitución de librerías de otros autores por código propio aumenta el rendimiento de algunas funciones vitales para el proyecto, ya que es mucho más reducido, esto mejora la velocidad de transferencia de dicho código entre el cliente y servidor.

8.2. Trabajos futuros.

Al concluir la primera implantación del sistema se procederá a agregar más módulos que integren al SITM la gestión de la biblioteca, tutorías y los exámenes en línea.

Mediante la utilización de alguna técnica de Inteligencia Artificial se propone realizar la generación automática de horarios. La estructura actual de la base de datos permite crear matrices de restricciones duras y blandas para optimizar horarios flexibles, que cumplan las restricciones duras y optimicen las restricciones blandas. Entre algunas de las técnicas de IA se proponen las siguientes basándose en algunos de muchisimos ejemplos dentro la literatura con relación al estudio de este campo:

Sistema MultiAgente: Este tipo de sistemas es muy utilizado para dar solución a distintos problemas de logística, está compuesto por agentes cooperantes que permiten un procesamiento altamente distribuido del problema con un gran potencial en el área de generación de horarios. Un ejemplo de esto se presenta en la obra "A Multi-agent Model for the University Timetabling Problem" donde implementan un modelo que satisfaga las limitaciones duras y blandas mientras que reduce la complejidad temporal. Para el análisis de la eficiencia del modelo, mostraron los resultados experimentales basados en instancias reales de la Higher Business School de Túnez analizando el efecto de variación de la clase magistral sobre el número de mensajes y el tiempo de ejecución del CPU y el efecto de la variación de la puntuación de prioridad de asignación en el porcentaje de satisfacción de las preferencias del profesor, (Cruz, 2016).

Colonia de hormigas: El algoritmo de Colonia de Hormigas es una metaheurística capaz de encontrar soluciones de buena calidad a problemas de optimización altamente complejos, se basa en la habilidad que poseen las hormigas naturales para encontrar el alimento a partir de individuos relativamente simples, pero con una estructura social muy eficiente. En el artículo "Herramientas heurísticas para la asignación optima de horarios de clase" proponen una solución basada en colonia de hormigas para el problema que enfrentan las instituciones educativas con respecto a la asignación optima de aulas. Para lograr el objetivo usaron las siguientes variables como factor de decisión: el número de eventos programados en aulas, el total de estudiantes inscritos a la institución, número de estudiantes por evento y total de aulas disponibles. En la etapa de resultados no se menciona algún porcentaje alcanzado, solo se indica que se encontró una solución óptima en base a las pruebas que realizaron, (Jaillivi Marín Lozada, 2013).

Algoritmos genéticos: Son métodos de búsqueda basados en el mecanismo de selección natural, lo que permite que se explore en busca de posibles soluciones en un espacio de búsqueda mayor al espacio de búsqueda de los métodos tradicionales. Se debe comenzar con un grupo de cromosomas o solución inicial que generalmente es obtenida de forma aleatoria o a través de otro método y se define una función objetivo que se encarga de entregar una medida de bondad de la solución. El principal objetivo en la obra "Generación de horarios académicos en INACAP utilizando algoritmos genéticos" consistió en construir una solución que ayudara a reducir el tiempo destinado a la creación de horarios académicos en INACAP, antes de desarrollar la herramienta de generación a través de un algoritmo genético ese proceso tardaba semanas o días en el mejor de los casos pero después de su implementación tan solo tomaba minutos u horas, además, el algoritmo encuentra soluciones con 90 % de secciones sin problemas de choques horarios y en poco más de 20 generaciones encuentra una solución óptima, lo cual representa un sustancial ahorro de recursos, (Ahumada, 2014).

Referencias

Ahumada, J. A. A. (2014). *Generación de horarios académicos en INACAP utilizando algoritmos genéticos* (Tesis de Master no publicada). Universidad de Chile, Facultad De Ciencias Físicas y Matemáticas departamento de Ciencias de la Computación.

Alonso Vega, A. (2013). *Responsive web design interfaces web adaptables al dispositivo empleando html5 y css3* (Tesis Doctoral, Universidad de Alcalá. Escuela Politécnica Superior). https://ebuah.uah.es/dspace/handle/10017/19972.

Ankurkar, S., y Sable, D. M. (2016). Evolving ajax with json for web application enrichment. *International Journal of Computer Applications, International Conference on Emerging Trends in Computing*.

Bahit, E. (2011). *El paradigma de la programación orientada a objetos en php con el patrón arquitectónico mvc.*

Boehm, B. W. (1988). A spiral model of software development and enhancement. *IEEE Computer Society*, *21*, 61 - 72.

Caballero, J. J. V. (2016). *Modelo de procesos para el desarrollo del front-end de aplicaciones web.* http://alicia.concytec.gob.pe/vufind/Record/RULI_c3d0a75d403c26d921004c6248c07423. INTERFASES.

Castejón, J. S. (2004). Arquitectura y diseño de sistemas web modernos. *InforMAS*, *1*, 2-3.

Cruz, O. L. (2016). Matp: A multi-agent model for the university timetabling problem. *Springer International Publishing Switzerland*, 11-22.

de Guevara, L. V. (2018). *Gestión de bases de datos.* https://media.readthedocs.org/pdf/gestionbasesdatos/latest/gestionbasesdatos.pdf.

del Carmen Gómez Fuentes, M. (2013). *Notas del curso: Bases de datos* (1ra Edición ed.). Casa abierta al tiempo, Universidad Autónoma Metropolitana.

Ecma, I. (2017). *Standard ecma 404: The json data interchange syntax.* Descargado Online; accesado el 31 de agosto de 2017, de https://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

ecma international.org. (2015). *Ecmascript 6.* Descargado Online; accesado el 15 de febrero de 2018, de https://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf

España León, A. R., Gonzáles Valero, M. I., Mejía Viter, J. T., Campi Mayorga, J. A., y Campi Mayorga, I. I. (2016). "patrón mvc, un componente para la implementación de una estrategia informática para mejorar gestión de datos en el área de estadística:

Caso de estudio hospital maternidad babahoyo.". *Revista de Ciencia, Tecnología e Innovación*, *3*, 31-41.

Fernández, D. Y., Y. (2012). Patrón modelo-vista-controlador. *Revista Telem@tica*.

IEEE. (2001). *Ansi/ieee 1471-2000 - ieee recommended practice for architectural description for software-intensive systems.* Descargado Online; accesado el 21 de febrero de 2018, de https://standards.ieee.org/findstds/standard/1471-2000.html

Jaillivi Marín Lozada, C. A. P. J. J. S. C., Diana Lorena Hoyos. (2013). Herramientas heurísticas para la asignación optima de horarios de clase. *Investigación en Ingeniería*, *10*, 68-74.

Johanna M. Suárez, L. E. G. (2015). *Tipificación de dominios de requerimientos para la aplicación de patrones arquitectónicos.* https://scielo.conicyt.cl/scielo.php?script=sci_arttext&pid=S0718-07642016000400021.

Json.org. (2013). *Introducción a json.* Descargado Online; accesado el 15 de febrero de 2018, de https://www.json.org/json-es.html

Leprohon Marc, A. (2017). *Ecmascript 6 and the evolution of javascript: A deeper look into the language's new features* (Tesis Doctoral, Helsinki Metropolia University of Applied Sciences, Bachelor of Engineering). https://www.theseus.fi/handle/10024/128511.

Marcotte, E. (2011). *Responsive web design* (2da Edition ed.). A Book Apart.

Martínez Villalobos, G., Camacho Sánchez, G. D., y Biancha Gutiérrez, D. A. (2010). Diseño de framework web para el desarrollo dinámico de aplicaciones. *Scientia et technica*.

mozilla. (2012). *Ajax.* Descargado Online; accesado el 21 de febrero de 2018, de https://developer.mozilla.org/es/docs/Web/Guide/AJAX

Nieto Lemus, A. C. (2015). "arquitectura por componentes jee, un caso práctico". *Revista Gerencia Tecnológica Informática*, *14*, 1-14.

Nurseitov, N., Paulson, M., Reynolds, R., y Izurieta, C. (2009). Comparison of json and xml data interchange formats: A case study.

Petković, D. (2017). Json integration in relational database systems. *International Journal of Computer Applications*.

php.net. (2018). *¿qué es php?* Descargado Online; accesado el 21 de febrero de 2018, de http://php.net/manual/es/intro-whatis.php

Popel D., C. L. (2007). *Learning php data objects: A beginner's guide to php data objects, database connection abstraction library for php5.*

Pressman, R. S. (2010). *Ingeniería del software* (7ma Edition ed.). McGraw-Hill.

Rivera Silva, A. C., Vargas Reyes, R. E., y Bohórquez Arévalo, L. E. (2018). "implementación de recursos empresariales (erp) en las organizaciones desde la coevolución". *Ingeniería Solidaria*, *14*, 1-22.

Sommerville, I. (2005). *Ingeniería del software* (7ma Edition ed.). Pearson Addison Wesley.

Tabarés Gutiérrez, R. (2016). El surgimiento de html5; un nuevo paradigma en los estándares web. *Teknokultura*.

Valzacchi, J. R. (2003). *Internet y educación aprendiendo y enseñando en los espacios virtuales.* INTERAMER.

W3C. (2016). *Html and css.* Descargado Online; accesado el 21 de febrero de 2018, de https://www.w3.org/standards/webdesign/htmlcss

Woychowsky, E. (2008). *Ajax creating web pages with asynchronous javascript and xml.*

```
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4  <meta charset="UTF-8">
5  <link rel="stylesheet" href="../plantilla/css/sitm.css">
6  <link rel="stylesheet" href="../plantilla/css/materias.css">
7  <link rel="stylesheet" href="../plantilla/fonts/style.css">
8  <link rel="stylesheet" href="../plantilla/css/mensajes.css">
9  <link rel="stylesheet" href="../plantilla/css/ventana.css">
10 <link rel="stylesheet" href="css/seleccionCurso.css">
11 <link rel="stylesheet" href="css/ev_materias.css">
12 <link rel="stylesheet" href="css/kardex.css">
13 <link rel="stylesheet" href="css/horario.css">
14 <link rel="stylesheet" href="css/adeudoAlumno.css">
15 <script src="../plantilla/js/sitm.js"></script>
16 <script src="../plantilla/js/mensajes.js"></script>
17 <script src="../plantilla/js/menu.js"></script>
18 <script src="../plantilla/js/menuItem.js"></script>
19 <script src="../plantilla/js/tabla.js"></script>
20 <script src="../plantilla/js/registro.js"></script>
21 <script src="../plantilla/js/campo.js"></script>
22 <script src="../plantilla/js/materia.js"></script>
23 <script src="../plantilla/js/materias.js"></script>
24 <script src="../plantilla/js/bloqueMaterias.js"></script>
25 <script src="../plantilla/js/utils.js"></script>
26 <script src="../plantilla/js/Ajax.js"></script>
27 <script src="../plantilla/js/gdata.js"></script>
28 <script src="js/dataTablaKardex.js"></script>
29 <script src="js/kardex.js"></script>
30 <script src="js/seleccionCurso.js"></script>
31 <script src="js/itemSeleccionCurso.js"></script>
32 <script src="js/datosExtraidosKardex.js"></script>
33 <script src="js/headKardex.js"></script>
34 <script src="js/TablaKardex.js"></script>
35 <script src="js/index_ev.js"></script>
36 <script src="js/index.js"></script>
37 <script src="js/funcionesCargaAcademica.js"></script>
38 <script src="js/funcionesKardex.js"></script>
39 <script src="js/ev_materias.js"></script>
40 <script src="js/horarios.js"></script>
41 <script src="js/tablaAdeudos.js"></script>
42 <script src="js/funcionesAdeudos.js"></script>
43 <title>Sistema Integral del ITSM: Alumnos</title>
44 </head>
45 <body>
46 </body>
47 </html>
```

Índice de algoritmos 8.1: Código de index.html.

```
*{
  margin: 0px;
  padding: 0px;
  border: 0px;
  font-family: Roboto, arial, sans-serif;
}

html, body {
  height: 100%;
  overflow: hidden;
}

body{
  display: flex;
  flex-direction: column;
  width: 100%;
}

header{
  flex-grow: 0;
  width: 100%;
  height: auto;
  border-bottom: 1px solid #e8e8e8;
  line-height: 0px;
  background:#fff;
}
header img{
  width: 100%;
}

header h1{
  display: none;
}

header span.opciones{
  display: none;
  position: relative;
}

#principal{
  display: flex;
  flex-direction: row-reverse;
  flex-grow: 1;
  height: 80%;
}

#area{
  display: flex;
  width: 100%;
  min-height: 100%;
```

```
51    flex-direction: column;
52 }
53
54 nav.menu{
55    height: auto;
56    width: 100%;
57 }
58
59 ul#menu {
60    height: auto;
61    display: flex;
62    align-items: center;
63    border-bottom: 3px solid #e8e8e8;
64    background: #fff;
65    justify-content: center;
66    flex-flow:row wrap;
67 }
68
69 ul#menu li{
70    width: 150px;
71    margin: 0px 10px;
72    padding: 3px 0;
73    text-align: center;
74    display: inline-block;
75    height: 45px;
76    font-size: .9rem;
77 }
78
79 ul#menu li:hover{
80    background: #e8e8e8;
81    cursor: pointer;
82 }
83
84 ul#menu li span{
85    font-size: 1.8rem;
86    line-height: 1px
87 }
88
89 ul#menu li span:before{
90    line-height: 25px;
91 }
92
93
94 ul#menu li span:after{
95    content: "\a";
96    white-space: pre;
97 }
98
99 aside.panel{
100   transition-duration: .5s;
101   width: 0px;
```

```
102     display: none;
103     min-width: 0px;
104     border-right: 1px solid #e8e8e8;
105     background: #fff;
106     overflow: scroll;
107     overflow-x: hidden;
108 }
109 
110 aside::-webkit-scrollbar{
111     width:8px;
112     background-color:rgba(0,0,0,0);
113 }
114 
115 aside::-webkit-scrollbar-thumb{
116     background-color:rgba(0,0,0,0.3);
117 }
118 
119 aside::-webkit-scrollbar-thumb:hover{
120     background-color:rgba(0,0,0,0.5);
121 }
122 
123 #lateral{
124     margin: 20px 25px;
125     text-align: justify;
126     width: 180px;
127     height: auto;
128 }
129 #lateral h3{
130     border-top: 2px solid #e8e8e8;
131     height: 32px;
132     width: 176px;
133     margin: 0px auto;
134     font-size: .7rem;
135     color: red;
136     position: relative;
137     display: flex;
138     align-items: center;
139 }
140 #lateral h3 span.icon-angle-down{
141     display: none;
142 }
143 #lateral h3 span{
144     height: auto;
145     width: auto;
146     font-size: 2em;
147     color: red;
148     position: absolute;
149     top: 4px;
150     right: 10px;
151 }
152 
```

```
153 #lateral h3:first-child{
154     width: 170px;
155     background: #6F767F;
156     padding-left: 10px;
157     color:#fff;
158 }
159 
160 #lateral h3:first-child span{
161     font-size: 2em;
162     color: #fff;
163     position: absolute;
164     top: 4px;
165     right: 10px;
166 }
167 
168 #lateral ul{
169     overflow: hidden;
170     transition-duration: .5s;
171 }
172 
173 #lateral ul li{
174     width: 166px;
175     height: auto;
176     padding: 6px 10px;
177     text-align: left;
178     margin: 0px auto;
179     font-size: .8rem;
180     display: flex;
181     align-items: center;
182 }
183 
184 #lateral ul li:not(first-child):hover{
185     color: #fff;
186     cursor: pointer;
187     background: rgb(68,68,68);
188 }
189 
190 #lateral ul li i, #lateral ul span{
191     font-size: 1.1rem;
192     margin-right: 10px;
193     height: auto;
194     width: auto;
195 }
196 
197 section{
198     overflow-y:auto;
199     background: #e8e8e8;
200     flex-grow: 1;
201 }
202 
203 section::-webkit-scrollbar{
```

```
204     width:8px;
205     background-color:rgba(0,0,0,0);
206 }
207
208 section::-webkit-scrollbar-thumb{
209     background-color:rgba(0,0,0,0.3);
210 }
211
212 section::-webkit-scrollbar-thumb:hover{
213     background-color:rgba(0,0,0,0.5);
214 }
215
216 @media screen and (max-width: 1250px) {
217     header{
218         min-height: 64px;
219         height: 64px;
220         max-height: 64px;
221     }
222
223     header span.opciones{
224         display: inline-block;
225         font-size: 64px;
226         height: 64px;
227     }
228
229     header img{
230         position: absolute;
231         height: 64px;
232     }
233
234
235     aside.panel{
236         z-index: 10;
237         left: 0px;
238         position: fixed;
239         height:100%;
240     }
241 }
242
243
244 @media screen and (max-width: 900px) {
245
246     header{
247         display: flex;
248         align-items:center;
249         min-height: 50px;
250         height: 50px;
251         max-height: 50px;
252     }
253
254     header span.opciones{
```

```
255       font-size: 50px;
256       height: 50px;
257     }
258
259     header img{
260       height: 50px;
261     }
262   }
263
264   @media screen and (max-width: 725px) {
265     header{
266       background: #fafafa;
267     }
268
269     header img{
270       height: 35px;
271     }
272
273     aside.panel{
274       margin-top: 0px;
275       height:calc(100% - 38px);
276     }
277   }
278
279   @media screen and (max-width: 520px) {
280     html{
281       height: auto;
282       background: #e8e8e8;
283       overflow-y: auto;
284     }
285
286     header{
287       z-index: 9;
288     }
289
290     header span.opciones{
291       display: inline-block;
292       position: fixed;
293       height: 50px;
294     }
295
296     header img{
297       display: none;
298     }
299
300     header h1{
301       width: 100%;
302       height: 50px;
303       font-size: .9rem;
304       display: flex;
305       align-items: center;
```

```
306       justify-content: center;
307       position: fixed;
308       background-color: #fff;
309       border-bottom: 1px solid #e8e8e8;
310     }
311
312     #area{
313       display: block;
314       z-index:5;
315     }
316
317     aside.panel{
318       height:calc(100% - 40px);
319     }
320
321     nav.menu{
322       border-bottom: 1px solid #e8e8e8;
323       height: auto;
324       position:static;
325     }
326
327     ul#menu {
328       align-items: initial;
329       height: auto;
330     }
331
332     ul#menu li{
333       margin-bottom: 10px;
334       border: 0px;
335       height: auto;
336     }
337
338     ul#menu li:hover{
339       margin-bottom: 10px;
340       border: 0px;
341     }
342
343     ul#menu li{
344       width: 100px;
345     }
346
347     section{
348       flex-grow: 2;
349       overflow-y: hidden;
350     }
351 }
```

Índice de algoritmos 8.2: Código CSS3 de la plantilla

```
1  class mensajes{
2    alerta(msg, fn = this.fnTrue, aceptar = "Aceptar"){
3      this.wrapper();
4      this.msg[this.msg.length-1].innerHTML+='<div id="msgEncabezado"><h4>'+this
       .hd+'</h4></div><div id="msgTexto">'+msg+'</div><div id="msgEntrada"></div
       ><div id="botones"><input type="button" id="msgAceptar" autofocus="" value
       ="'+aceptar+'" class="clickeable"></div>';
5      this.eventos(fn);
6    }
7
8    confirmar(msg, fn = this.fnTrue, fnNo = this.fnTrue, aceptar = "Aceptar",
       cancelar = "Cancelar"){
9      this.wrapper();
10     this.msg[this.msg.length-1].innerHTML+='<div id="msgEncabezado"><h4>'+this
       .hd+'</h4></div><div id="msgTexto">'+msg+'</div><div id="botones"><input
       type="button" id="msgAceptar" value="'+aceptar+'" class="clickeable"><
       input type="button" id="msgCancelar" autofocus="" value="'+cancelar+'"
       class="clickeable"></div>';
11     this.eventos(fn, fnNo);
12   }
13
14   confirmar_sinCancelar(msg, fn = this.fnTrue, txt = aceptar, aceptar = "
       Aceptar"){
15     this.wrapper();
16     this.msg[this.msg.length-1].innerHTML+='<div id="msgEncabezado"><h4>'+this
       .hd+'</h4></div><div id="msgTexto">'+msg+'</div><div id="msgEntrada"></div
       ><div id="botones"><input type="button" id="msgAceptar" autofocus="" value
       ="'+txt+'" class="clickeable"></div>';
17     this.eventos(fn);
18   }
19
20   capturar(msg, fn = this.fnTrue, fnNo = this.fnTrue, valor = "", aceptar = "
       Aceptar", cancelar = "Cancelar"){
21     this.wrapper();
22     this.msg[this.msg.length-1].innerHTML+='<div id="msgEncabezado"><h4>'+this
       .hd+'</h4></div><div id="msgTexto">'+msg+'</div><div id="msgEntrada"><
       input type="text" id="msgValor" autofocus="" value="'+valor+'"></div><div
       id="botones"><input type="button" id="msgAceptar" value='+aceptar+' class
       ="clickeable"><input type="button" id="msgCancelar" autofocus="" value
       ="'+cancelar+'" class="clickeable"></div>';
23     this.eventos(fn, fnNo);
24   }
25
26   capturar_sinCancelar(msg, fn = this.fnTrue, valor = "", aceptar = "Aceptar")
       {
27     this.wrapper();
28     this.msg[this.msg.length-1].innerHTML+='<div id="msgEncabezado"><h4>'+this
       .hd+'</h4></div><div id="msgTexto">'+msg+'</div><div id="msgEntrada"><
       input type="text" id="msgValor" autofocus=""></div><div id="botones"><
       input type="button" id="msgAceptar" autofocus="" value='+aceptar+' class="
```

```
    clickeable"></div>';
29      this.eventos(fn);
30    }
31
32    wrapper(){
33      let ms = document.querySelector('#mensajes'), m = document.createElement('
        div');
34      if(ms == null){
35        ms = document.createElement('div');
36        ms.setAttribute("id", "mensajes");
37        document.body.insertBefore(ms, document.body.children[0]);
38      }
39      m.setAttribute("id", "mensaje");
40      m.innerHTML = '<div id="cerrar"><svg id="msgCerrar" xmlns="http://www.w3.
        org/2000/svg" width="30px" height="30px" viewBox="0 0 612 612"><polygon
        points="612,36.004 576.521,0.603 306,270.608 35.478,0.603 0,36.004
        270.522,306.011 0,575.997 35.478,611.397 306,341.411 576.521,611.397
        612,575.997 341.459,306.011" fill="#FFF"/></svg></div><div id="caja"></div
        >';
41      if(document.querySelector("#mensaje") == null) ms.insertBefore(m, ms.
        children[0]);
42      else{
43        let mc = document.querySelectorAll("#mensaje>#caja"), l = mc.length;
44        ms.insertBefore(m, ms.children[l]);
45      }
46      this.msg = document.querySelectorAll("#mensaje>#caja");
47      this.hd = "Sistema Integral del TecNM";
48    }
49
50    eventos(fn = this.fnTrue, fnNo = this.fnTrue){
51      let c = document.querySelector("#msgCancelar"),
52      a = document.querySelector("#msgAceptar"),
53      v = document.querySelectorAll("#mensaje input[type=text]"),
54      mC = document.querySelectorAll("#msgCerrar");
55
56      for (var cl = 0; cl < mC.length; cl++)
57      mC[cl].onclick = (e) => mensajes.cerrar();
58      a.onclick = (e) => {
59        if(v.length == 1) v = document.querySelector("#msgValor");
60        else if(v.length > 1){
61          let valores={}
62          for (var i = 0; i < v.length; i++)
63          Object.defineProperty(valores, v[i].id, {value: v[i].value, writable:
        true, enumerable:true, configurable:true});
64          v = valores;
65        }
66        else v = null;
67        mensajes.cerrar();
68        if(v != null && v.hasOwnProperty("value")) fn(e, v != null ? v.value :
        "");
69        else fn(e, v != null ? v : "");
```

```
70      };
71      if(c != null) c.onclick = (e) => { mensajes.cerrar(); fnNo(e); }
72      document.body.addEventListener('keydown', mensajes.teclas);
73    }
74
75    fnTrue() { return true };
76
77    static teclas(e){
78      let k = e.keyCode;
79      if(k == 13){
80        let a=document.querySelector("#msgAceptar"),
81        c= document.querySelector("#msgCancelar"),
82        t =document.querySelector("#msgValor");
83        if (t!=null) t.value!="" ? a.click() : c.click();
84        else c!=null ? c.click() : a.click();
85      }
86      else if(k == 27) mensajes.cerrar();
87      e.preventDefault();
88    }
89
90    static cerrar(){
91      let m = document.querySelectorAll("#mensaje"), l = m.length;
92      m = m[l-1];
93      m.parentNode.removeChild(m);
94      document.body.removeEventListener('keydown', mensajes.teclas);
95    }
96  }
```

Índice de algoritmos 8.3: Código de la clase mensajes.

```
1  class tabla {
2    constructor(registros, setup = null){
3      let filas = '';
4      this.html = '<div ';
5      if(setup !== null){
6        if(setup.hasOwnProperty('class'))
7          this.html += 'class="' + setup.class + '" ';
8        if(setup.hasOwnProperty('id'))
9          this.html += 'id="' + setup.id + '"';
10     }
11     for (let i = 0; i < registros.length; i++)
12       filas += new registro(registros[i].data, registros[i].hasOwnProperty('
       setup') ? registros[i].setup : null).html;
13     this.html += '>' + filas + '</div>';
14   }
15 }
```

Índice de algoritmos 8.4: Código de la clase tabla.

```
class registro{
  constructor (campos, setup = null) {
    var fields = '';
    this.html = '<div ';
    if(setup !== null){
      if(setup.hasOwnProperty('class'))
        this.html += 'class="' + setup.class + '" ';
      if(setup.hasOwnProperty('id'))
        this.html += 'id="' + setup.id + '"';
    }
    for (var i = 0; i < campos.length; i++)
      fields += new campo(campos[i].data, campos[i].hasOwnProperty('setup') ?
   campos[i].setup : null).html;
    this.html += '>' + fields + '</div>';
  }
}
```

Índice de algoritmos 8.5: Código de la clase registro.

```
class campo{
  constructor(data, setup = null){
    this.html = '<span ';
    if(setup !== null){
      if(setup.hasOwnProperty('class'))
        this.html += 'class="' + setup.class + '" ';
      if(setup.hasOwnProperty('id'))
        this.html += 'id="' + setup.id + '"';
    }
    this.html += '>' + data + '</span>';
  }
}
```

Índice de algoritmos 8.6: Código de la clase campo.

```
class kardex{
  constructor(){
    let materias = kardex.ordenarMaterias()[0],
      reticula = gd.html_getItem("materias"),
      plan = gd.html_getItem("perfil").carreras[0],
      personales = gd.html_getItem("personales")[0],
      carreras = gd.html_getItem("carreras"),
      pctje, carrera, cal ,
      creTo = 0 , creAc = 0 , creCu = 0 , totalMat = 0 ,
      sumaCalSR = 0 , sumaCalCR = 0 , promSinR = 0 , promConR = 0 ,
      acreditadas = 0 , cursadas = 0 , aprobadas = 0 ,
      sinConsiderar = materias.sinConsiderar;
    delete materias.sinConsiderar;
    for(let materia in materias){
      if(materias[materia]!="-")
        totalMat++;
```

```
17        if(materias[materia].CURSOS != undefined){
18          let op1=materias[materia].CURSOS[0],
19            op2=materias[materia].CURSOS[1],
20            op3=materias[materia].CURSOS[2];
21          if(op1.CA == "ACR" || op2.CA == "ACR" || op3.CA == "ACR")
22            cal = "ACR";
23          else
24            cal = Math.max(op1.CA , op2.CA , op3.CA);
25          if(cal != -1 && cal != "ACR")
26            sumaCalCR = sumaCalCR + cal;
27          if(cal >= 70 || cal == "ACR"){
28            aprobadas++;
29            if(cal != "ACR")
30              sumaCalSR = sumaCalSR + cal;
31            else
32              acreditadas++;
33          }
34          if(op1.PE != "")
35            cursadas++;
36        }
37        for(let cd in reticula){
38          if(reticula[cd].clave == materia){
39            creTo = creTo + reticula[cd].creditos;
40          if(cal >=70 || cal == "ACR")
41            creAc = creAc + reticula[cd].creditos;
42          if(materias[materia].CU == 1)
43            creCu = creCu + reticula[cd].creditos;
44          }
45        }
46      }
47      promSinR = sumaCalSR / (aprobadas-acreditadas);
48      promSinR = promSinR.toFixed(2);
49      promConR = sumaCalCR / (cursadas-acreditadas);
50      promConR = promConR.toFixed(2);
51      pctje = creAc*100/creTo;
52      pctje = pctje.toFixed(0);
53
54      for(let i in carreras)
55        if(carreras[i].plan == plan)
56          carrera = carreras[i].descripcion;
57
58      let size = Object.keys(materias).length - 1,
59        inicio = Object.keys(materias)[sinConsiderar],
60        fin = Object.keys(materias)[size],
61        p1pm = materias[inicio].CURSOS[0].PE,
62        p2pm = materias[inicio].CURSOS[1].PE,
63        p3pm = materias[inicio].CURSOS[2].PE;
64      if(p3pm != "")
65        inicio = p3pm;
66      else if (p2pm != "")
67        inicio = p2pm;
```

```
68      else
69        inicio = p1pm;
70      if(materias[fin].CURSOS != undefined){
71        let op1um = materias[fin].CURSOS[0],
72          op2um = materias[fin].CURSOS[1],
73          op3um = materias[fin].CURSOS[2];
74        if(op1um.CA == "ACR" || op2um.CA == "ACR" || op3um.CA == "ACR")
75          cal = "ACR";
76        else
77          cal = Math.max(op1um.CA , op2um.CA , op3um.CA)
78        if(cal >=70 || cal =="ACR"){
79          if(op3um.PE != "")
80            fin = op3um.PE;
81          else if (op2um.PE != "")
82            fin = op2um.PE;
83          else
84            fin = op1um.PE
85        }
86        else fin = "";
87      }
88      let data = { nControl: personales.login , nombre : {nm : personales.nombre
        .nm , pa : personales.nombre.pa , sa : personales.nombre.sa} , carrera :
        carrera , sem : personales.semestre ,
89       sit: personales.situacion, plan : plan , ingr : inicio, term : fin ,
        totalMat : totalMat , acreditadas : acreditadas , aprobadas : aprobadas ,
        cursadas : cursadas , pctje :pctje , creTo : creTo ,
90       creAc : creAc , creCu : creCu , promSinR : promSinR , promConR : promConR
        , materias : materias};
91       return data;
92    }
93
94    static ordenarMaterias(){
95      let materias = gd.html_getItem("kardex")[0],
96        reticula = gd.html_getItem("materias"),
97        lista = [] , orden = [] , periodos = [], ordenar= [],
98        ordenado, nombre, periodo, sinConsiderar = 0;
99      delete materias.nControl;
100     for(let materia in materias){
101       if(materias[materia].CURSOS!=undefined){
102         let pe = materias[materia].CURSOS[0].PE;
103         if(pe!=""){
104           let mes = pe.substr(0, 3),
105           anio = parseInt(pe.substr(3, 2));
106           if(isNaN(anio))
107           anio = parseInt(pe.substr(-2, 2));
108           if(mes=="ENE")
109             mes = 1
110           else if(mes=="FEB")
111             mes = 2
112           else if(mes=="VER")
113             mes = 3
```

```
114             else if(mes=="AGO")
115               mes = 4
116             else{}
117             periodo = anio+""+mes;
118           }else
119             periodo = "-10";
120           for(let m in reticula)
121             if(m.clave == materia)
122               nombre = m.nombre;
123
124         lista.push({clave : materia , nombre : nombre, periodo : periodo});
125         }
126       }
127       for(let li in lista){
128         orden.push(lista[li]);
129         periodos.push(lista[li].periodo);
130       }
131       periodos = Array.from(new Set(periodos));
132       periodos = periodos.sort(function(a, b){ return (a - b)});
133       for (let periodo in periodos) {
134         let segmentoPeriodo = [], segmentoOrdenado;
135         for(let item in orden)
136           if(orden[item].periodo==periodos[periodo])
137             segmentoPeriodo.push(orden[item]);
138
139         segmentoOrdenado = segmentoPeriodo.sort(function(a, b){
140           if(a.nombre < b.nombre) return -1;
141           if(a.nombre > b.nombre) return 1;
142           return 0;
143         })
144
145         for(let sp in segmentoOrdenado)
146           ordenar.push(segmentoOrdenado[sp].clave);
147
148         if(periodos[periodo]==-10)
149         sinConsiderar=segmentoOrdenado.length;
150       }
151
152       ordenado ='[{';
153       for(let materia in materias){
154         if(materias[materia].CURSOS == undefined){
155           if(materias[materia]=="-")
156             ordenado += '"'+ materia +'":"-",';
157           else
158             ordenado += '"'+ materia +'":{},';
159           sinConsiderar++;
160         }
161       }
162       for(let orden in ordenar){
163         let materia = gd.html_getItem("kardex")[0][ordenar[orden]];
164         ordenado+= '"'+ordenar[orden] +'":'+ JSON.stringify(materia)+',';
```

```
165     }
166     ordenado +='"sinConsiderar":'+ sinConsiderar +'}]';
167     ordenado = JSON.parse(ordenado);
168 
169     return ordenado;
170   }
171 }
```

Índice de algoritmos 8.7: Código de la clase kardex.

```
1  class sitm{
2    constructor(nivel = 0, reset = false){
3      if(reset) {
4        if(!this.existePerfil()) window.location = "../plantilla/html/ilegal.
       html";
5        else new Ajax().post("select_accesoLegal", {usuario: this.pf.usuario.
       login}, this.fnLogin);
6      }
7    }
8 
9    existePerfil(){ // Verifico que exista el objeto perfil y tenga la
       estructura esperada
10     if(!sessionStorage.hasOwnProperty("perfil")) return false;
11     else {
12       this.pf = JSON.parse(sessionStorage.getItem("perfil"));
13       return this.pf.hasOwnProperty("usuario") && this.pf.hasOwnProperty("
       modulos") && this.pf.hasOwnProperty("permisos");
14     }
15   }
16 
17   fnLogin(r){
18     let l = JSON.parse(r);
19     if(!l.accesoLegal) window.location = "../plantilla/html/ilegal.html";
20     else{
21       window.ajax = new Ajax();
22       window.gd = new gdata();
23       window.msg = new mensajes();
24     }
25   }
26 
27   gdataReset(aplicacion=""){ // Ejecutada en el descendiente para que defina
       la defina la aplicacion
28     let Yaplicacion=0;
29     let h = document.documentElement;
30 
31     if(!h.hasOwnProperty("data")) h.data = {};
32     h.data.perfil = JSON.parse(sessionStorage.getItem("perfil"));
33     sessionStorage.removeItem("perfil");
34 
35     ajax.post("select_dateTime");
36     if (aplicacion!="") ajax.post("select_constantes", {modulo: aplicacion});
```

```
37    }
38
39    render(nivel, botonOpciones = true){ // Ejecutada en el cliente para definir
        el nivel
40      let b = document.body,
41      u = this.levels(nivel) + 'plantilla/img/enc_itsm.png',
42      btn = botonOpciones ? '<span class="opciones"><span class="icon-fi-list
        clickeable" id="botonOpciones"></i></span>' : "";
43
44      b.innerHTML += '<header id="header"><h1>Tec. Misantla</h1><img id="logo"
        src="' + u +'">' + btn + '</header>';
45      b.innerHTML += '<div id="principal"><div id="area"></div><aside class="
        panel"><div id="lateral"></div></aside></div>';
46      this.menuPrincipal(this.dataMenuPrincipal());
47      document.querySelector("#area").innerHTML += '<section id="contenido"></
        section>';
48      this.permisos();
49    }
50
51    dataMenuPrincipal(){} // El cuerpo lo define el descendiente
52    permisos(){} // El cuerpo lo define el descendiente
53
54    levels(level){
55      let str = "";
56      for (var i = 0; i < level; i++) str += "../";
57      return str;
58    }
59
60    menuPrincipal(m){
61      if(m!=null){
62        document.querySelector("#area").innerHTML += '<nav class="menu"></nav>';
        // Creo el nav en el body
63        for (let i = 0; i < m.length; i++){ // Defino los elementos del menu y
        los fijo
64          let item = new menu(m[i].nombre, m[i].items, menu.SUPERIOR);
65          document.querySelector(".menu").innerHTML+=item.html;
66        }
67      }
68    }
69
70    menuLateral(m){
71      setTimeout ("document.querySelector('.panel').style.display ='flex';", 0);
72      setTimeout ("document.querySelector('.panel').style.minWidth ='230px';
        document.querySelector('.panel').style.width ='230px';", 400);
73
74      this.borrarChildren("#lateral", 0);
75
76      let lateral = document.querySelector("#lateral");
77
78      for (let i = 0; i < m.length; i++){ // Creo menu lateral
79        let item = new menu(m[i].nombre, m[i].items, menu.LATERAL, m[i].id, m[i
```

```
    ].tipo);
80        lateral.innerHTML+=item.html;
81      }
82
83      for (let i = sitm.menus.length - 1; i >= 0; i--) sitm.menus.pop(); //
      Limpio el array de control de menus
84
85      for (let i = 0; i < m.length; i++) // Conservo la altura de los menus
86      sitm.menus.push({id:m[i].id, up: "up"+m[i].id, down: "down"+m[i].id,
      height: document.querySelector("#"+m[i].id).offsetHeight});
87
88      this.onClick(); // Asocio eventos click a los controles
89    }
90
91    borrarChildren(elemento, nivel){
92      let items = document.querySelector(elemento);
93      while (items.children[nivel])
94      items.removeChild(items.children[nivel])
95    }
96
97    onClick() {
98      let clicks = document.querySelectorAll('#lateral .clickeable');
99      for(let i = 0; i < clicks.length;i++)
100     clicks[i].onclick = (e) => new index().eventHandler(e);
101   }
102
103   marcarSuperior(id){
104     var menu=document.querySelectorAll('#area > .menu> #menu > li'),
105     target=document.querySelector('#'+id);
106
107     for(let i = 0; i < menu.length;i++)
108     menu[i].style.backgroundColor='#fff';
109     target.style.backgroundColor='#e8e8e8';
110     for(let i = 0; i < menu.length;i++){
111       menu[i].onmouseover = function() {
112         this.style.backgroundColor = target.style.backgroundColor = '#e8e8e8';
113       }
114       menu[i].onmouseout = function() {
115         this.style.backgroundColor ='transparent';
116         target.style.backgroundColor='#e8e8e8';
117       }
118     };
119   }
120
121   marcarLateral(id){
122     var menu=document.querySelectorAll('#lateral > ul:not(.fijo) > li'),
123     target=document.querySelector('#'+id);
124
125     for(let i = 0; i < menu.length;i++){
126       menu[i].style.backgroundColor='#fff';
127       menu[i].style.color ='#000';
```

```
128       };
129       target.style.backgroundColor = 'rgba(0,0,0, 0.73)';
130       target.style.color='#fff';
131       for(let i = 0; i < menu.length;i++){
132         menu[i].onmouseover = function() {
133           this.style.backgroundColor = 'rgba(0,0,0, 0.73)';
134           target.style.backgroundColor = '#000';
135           this.style.color = target.style.color = '#fff';
136         }
137         menu[i].onmouseout = function() {
138           this.style.backgroundColor ='transparent';
139           this.style.color = '#000';
140           target.style.backgroundColor = '#000';
141           target.style.color ='#fff';
142         }
143       };
144     }
145
146     marcarLateralFijo(id){
147       var target=document.querySelector('#'+id),
148       padre=target.parentNode.id,
149       menu=document.querySelectorAll('#'+padre+' > li');
150
151       for(let i = 0; i < menu.length;i++){
152         menu[i].style.backgroundColor='#FFF';
153         menu[i].style.color ='#000';
154       };
155       target.style.backgroundColor = 'rgba(0,0,0, 0.73)';
156       target.style.color='#fff';
157       for(let i = 0; i < menu.length;i++){
158         menu[i].onmouseover = function() {
159           this.style.backgroundColor = 'rgba(0,0,0, 0.73)';
160           target.style.backgroundColor = '#000';
161           this.style.color = target.style.color = '#fff';
162         }
163         menu[i].onmouseout = function() {
164           this.style.backgroundColor ='transparent';
165           this.style.color = '#000';
166           target.style.backgroundColor = '#000';
167           target.style.color ='#fff';
168         }
169       };
170     }
171
172     // ==================
173     // Manejador de eventos
174     // ==================
175     eventHandler(e){
176       e.preventDefault();
177       if(typeof e.currentTarget.attributes.id != 'undefined'){
178         let id = e.currentTarget.attributes.id.value.trim();
```

```
179
180       if (document.querySelector('#lateral > ul.fijo > #'+id)!=null) this.
      marcarLateralFijo(id);
181       else if (document.querySelector('#lateral > ul > #'+id)!=null) this.
      marcarLateral(id);
182       else if(document.querySelector('#area > .menu > #menu > #'+id)!=null)
      this.marcarSuperior(id);
183
184       if(!this.esClickDeMenu(id))
185       switch(id){
186         case 'botonOpciones':
187         this.resizeMenu();
188         break;
189         default:
190         return false;
191       }
192       if(id != 'botonOpciones') this.upDownMenu(this.datosMenu(id), id);
193     }
194     return true;
195   }
196
197   esClickDeMenu(id){
198     for (let i = 0; i < sitm.menus.length; i++)
199     if(sitm.menus[i].up == id || sitm.menus[i].down == id) return true;
200     return false;
201   }
202
203   resizeMenu(){ // Gestores de evento
204     if (document.querySelector("#lateral").offsetHeight>40) {
205       let p = document.querySelector(".panel");
206       if(p.style.width == "230px") p.style.width = p.style.minWidth = "0px";
207       else p.style.width = p.style.minWidth="230px";
208     }
209     else new mensajes().alerta("Seleccione primero una de las opciones para
      navegar.");
210   }
211
212   upDownMenu(o, id){
213     let ctrl = document.querySelector("#" + o.id);
214     if(id.substring(0,2) == "up") {
215       ctrl.style.height = "0px";
216       document.querySelector("#" + o.up).style.display = "none";
217       document.querySelector("#" + o.down).style.display = "block";
218     }
219     else{
220       ctrl.style.height = o.height + "px";
221       document.querySelector("#" + o.up).style.display = "block";
222       document.querySelector("#" + o.down).style.display = "none";
223     }
224   }
225
```

```
226   datosMenu(id){
227     for (let i = 0; i < sitm.menus.length; i++)
228     if(sitm.menus[i].up == id || sitm.menus[i].down == id) return sitm.menus[i
        ];
229     return undefined;
230   }
231
232   static windowResize(){
233     let p = document.querySelector(".panel");
234     if(window.innerWidth <= 1250 && p != null)
235     p.style.width = p.style.minWidth = "0px";
236     if(window.innerWidth > 1250 && p != null && p != 0)
237     p.style.width = p.style.minWidth = "230px";
238   }
239 }
240
241 // Para la gestion de los menus
242 sitm.menus = [];
243
244 window.addEventListener('resize', function(event){
245   sitm.windowResize();
246 });
```

Índice de algoritmos 8.8: Código de la plantilla SITM.

```
1  class index extends sitm{
2    constructor(nivel = 1, reset = false){
3      if(reset) {super(nivel, reset);}
4      else super();
5    }
6
7    fnLogin(r){
8      super.fnLogin(r);
9      new index().gdataReset();
10   }
11
12   gdataReset(){
13     super.gdataReset("alumnos");
14     let pf = gd.html_getItem("perfil");
15     ajax.post("select_bloques", {carrera: pf.carreras[0]}, new index().
       fnBloques);
16     this.render(1);
17     let clicks = document.querySelectorAll('.clickeable');
18     for(let i = 0; i < clicks.length;i++){
19       clicks[i].onclick = (e) => new index().eventHandler(e);
20     }
21   };
22   fnBloques(r){
23     let data,
24     pf = gd.html_getItem("perfil");
25     try{ data = JSON.parse(r)}
```

```
26      catch(e){
27        utils.debug("index.fnBloques", r, utils.err_parse);
28        return;
29      }
30      gd.html_setItem("bloques", data);
31      ajax.post("select_reticula", {carrera: pf.carreras[0]}, new index().
      fnReticula);
32    }
33    fnReticula(r){
34      let data,
35      pf = gd.html_getItem("perfil");
36      try{ data =  JSON.parse(r)}
37      catch(e){
38        utils.debug("index.fnReticula", r, utils.err_parse);
39        return;
40      }
41      gd.html_setItem("materias", data);
42      ajax.post("select_catalogos", {catalogos: ['_carreras']}, new index().
      fnCarreras);
43    }
44    fnCarreras(r){
45      let data,
46      pf = gd.html_getItem("perfil");
47      try{
48        data =  JSON.parse(r);
49        data = data._carreras;
50      }
51      catch(e){
52        utils.debug("index.fnCarreras", r, utils.err_parse);
53        return;
54      }
55      gd.html_setItem("carreras", data);
56      ajax.post("select_kardex", {carrera: pf.carreras[0], nControl: pf.usuario.
      login},new index().fnKardex);
57    }
58    fnKardex(r){
59      let data,
60      pf = gd.html_getItem("perfil");
61      try{ data =  JSON.parse(r)}
62      catch(e){
63        utils.debug("index.fnKardex", r, utils.err_parse);
64        return;
65      }
66      gd.html_setItem("kardex", data);
67      ajax.post("select_personales_min", {login: pf.usuario.login}, new index().
      fnPersonales);
68    }
69    fnPersonales(r){
70      let data,
71      pf = gd.html_getItem("perfil");
72      try{ data =  JSON.parse(r)}
```

```
73      catch(e){
74        utils.debug("index.fnPersonales", r, utils.err_parse);
75        return;
76      }
77      gd.html_setItem("personales", data);
78      ajax.post("cargas_select_carrera", {carrera: pf.carreras[0]}, new index().
        fnCarga);
79    }
80    fnCarga(r){
81      let data,
82      pf = gd.html_getItem("perfil");
83      try{ data =  JSON.parse(r)}
84      catch(e){
85        utils.debug("index.fnCarga", r, utils.err_parse);
86        return;
87      }
88      gd.html_setItem("carga", data);
89      ajax.post("select_docentes", {}, new index().fnDocentes);
90    }
91    fnDocentes(r){
92      let data,
93      pf = gd.html_getItem("perfil");
94      try{ data =  JSON.parse(r)}
95      catch(e){
96        utils.debug("index.fnDocentes", r, utils.err_parse);
97        return;
98      }
99      gd.html_setItem("docentes", data);
100     ajax.post("select_validaciones", {nControl:pf.usuario.login},new index().
        fnValidaciones);
101   }
102   fnValidaciones(r){
103     let data,
104     pf = gd.html_getItem("perfil");
105     try{ data =  JSON.parse(r)}
106     catch(e){
107       utils.debug("index.fnValidaciones", r, utils.err_parse);
108       return;
109     }
110     gd.html_setItem("validaciones", data);
111     let permisos = gd.html_getItem("validaciones")[0].permisos,
112     aplicado = gd.html_getItem("validaciones")[0].aplicado;
113     if(permisos!=undefined){
114       if(permisos.bMin==1 && aplicado==0)
115       gd.html_getItem("constantes").CREDITOS_MINIMO = 1;
116       if(permisos.bMax==1 && aplicado==0)
117       gd.html_getItem("constantes").CREDITOS_MAXIMO = Infinity;
118     }
119     ajax.post("grupos_select_carrera", {carrera: pf.carreras[0]}, new index().
        fnGrupos);
120   }
```

```
121   fnGrupos(r){
122     let data,
123     pf = gd.html_getItem("perfil");
124     try{ data =  JSON.parse(r)
125     }
126     catch(e){
127       utils.debug("index.fnGrupos", r, utils.err_parse);
128       return;
129     }
130     gd.html_setItem("grupos", data);
131     ajax.post("select_adeudos_alumno", {nControl:pf.usuario.login},new index()
      .fnAdeudos);
132   }
133   fnAdeudos(r){
134     let data,
135     pf = gd.html_getItem("perfil");
136     try{ data =  JSON.parse(r)
137     }
138     catch(e){
139       utils.debug("index.fnAdeudos", r, utils.err_parse);
140       return;
141     }
142     gd.html_setItem("adeudos", data);
143
144     let adeudosMaterial = false,
145     adeudo = gd.html_getItem("adeudos")[1];
146     for(let i in adeudo){
147       if(adeudo[i].recibio=='no recibido')
148       adeudosMaterial = true;
149     }
150     gd.html_setItem("adeudosMaterial", adeudosMaterial);
151     new funcionesCargaAcademica();
152   }
153 }
154 index.seleccionado = 1;
```

Índice de algoritmos 8.9: Código del index.

```
1 window.onload = function(e){
2   new index(1,true);
3 }
```

Índice de algoritmos 8.10: Código de modulo_ev.