Universidad de las Ciencias Informaticas

Facultad 2

Título: Marco de trabajo Xilema base Web 2.0

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informático

Autor: Wendy Trujillo Delgado

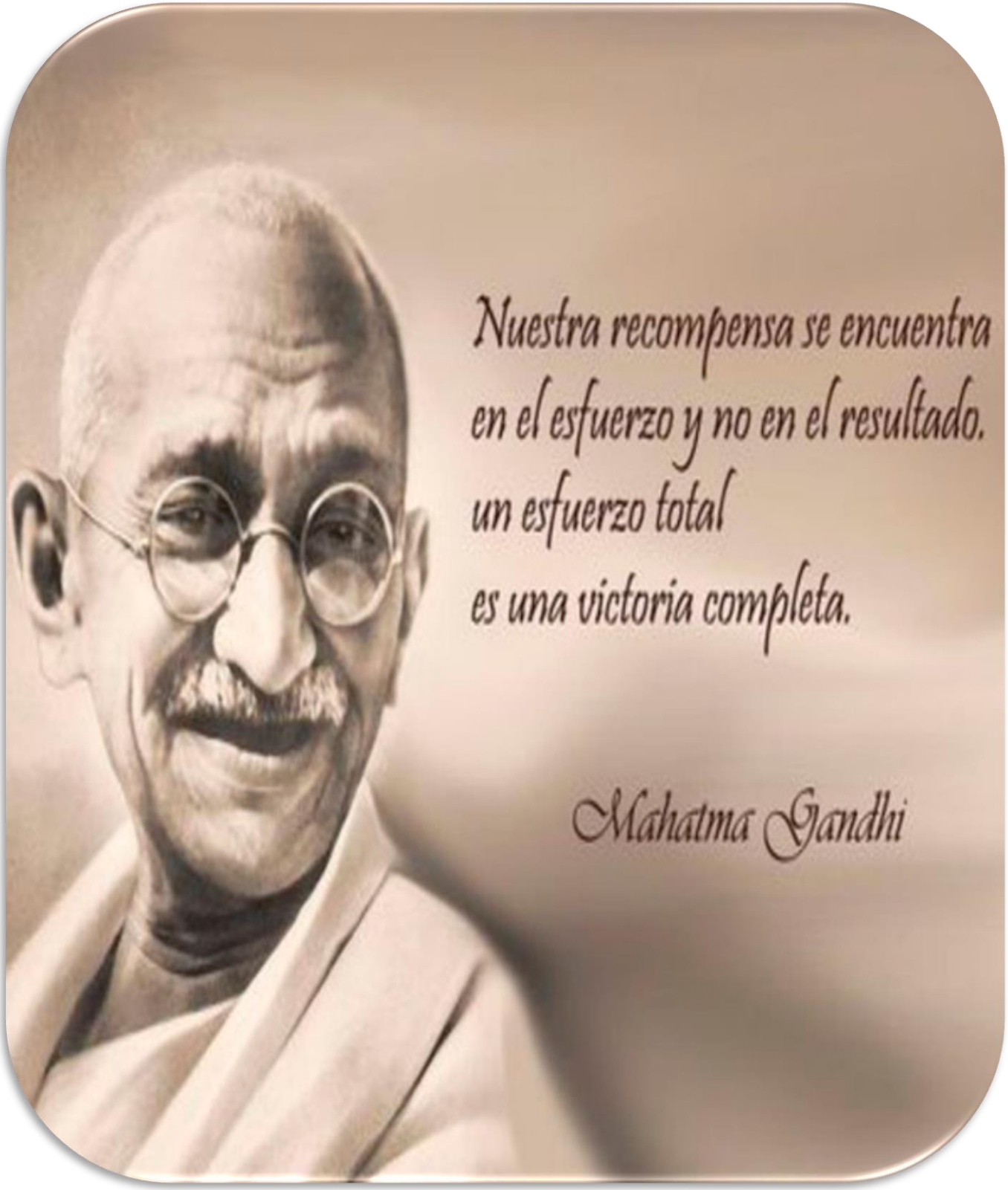
Tutores: MSc. Lianet Salazar Labrada

Ing. Jorge Armando Túñez González

Consultante: Ing. Andrés Escobar Pérez

La Habana, junio de 2018

"Año 60 de la Revolución"


Nuestra recompensa se encuentra
en el esfuerzo y no en el resultado.
un esfuerzo total
es una victoria completa.
Mahatma Gandhi

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis que tiene por título: Desarrollar el marco de trabajo Xilema base Web en su versión 2.0 y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los 11 días del mes de junio del año 2018.

Wendy Trujillo Delgado

_____

Firma del Autor

Lianet Salazar Labrada

_____

Firma del Tutor

Jorge Armando Túñez González

_____

Firma del Tutor

RESUMEN

En el Centro de Telemática perteneciente a la Facultad 2 de la Universidad de las Ciencias Informáticas se utiliza para el desarrollo de sus productos de software el marco de trabajo Xilema base Web 1.0, el cual implementa la estrategia marcaria Xilema de la universidad. La tecnología con que se desarrolló en sus inicios la interfaz de usuario de Xilema base Web actualmente se encuentra obsoleta, por lo que surge la necesidad de migrar dicha interfaz hacia nuevas tecnologías de desarrollo con el fin de mejorar la usabilidad y eficiencia del sistema, manteniendo las restricciones de diseño establecidas por la universidad. Para darle solución al problema se realizó un estudio de los marcos de trabajo que utilizan JavaScript en la actualidad, con el fin de seleccionar el más adecuado a las necesidades del sistema. El desarrollo de la migración de Xilema base Web a su versión 2.0 fue conducido por las especificaciones de la metodología de Proceso Unificado Ágil variación de la Universidad de las Ciencias Informáticas, obteniendo los artefactos generados por sus distintas fases de trabajo. Como propuesta de solución se establecieron dos servidores, uno orientado a la vista y las plantillas y el otro al modelo del sistema. Se utilizaron como nuevas tecnologías de desarrollo el marco de trabajo Angular y el entorno de desarrollo Bootstrap para el servidor encargado de la interfaz, así como, Python y Django como tecnologías para el servidor encargado de la lógica del sistema. Asimismo, se utilizaron otras herramientas y tecnologías en el trascurso del ciclo de vida del sistema tales como: Lenguaje Unificado de Modelado*,* Visual Paradigm*,* PostgreSQL, PyCharm, Visual Studio Code y Git.

PALABRAS CLAVE

Interfaz, JavaScript, marcos de trabajo, migrar, Xilema base Web.

ABSTRACT

In the Telematics Center belonging to the Faculty 2 of the University of Informatics Sciences, the Xilema base Web 1.0 framework is used for the development of its software products, which implements the university's Xilema design patterns. The technology with which the Xilema base Web user interface was developed at the beginning is currently obsolete, so the need arises to migrate this interface to new development technologies in order to improve the usability and efficiency of the system, maintaining the design restrictions established by the university. To solve the problem, a study was made of the frameworks that use JavaScript at present, in order to select the one that best suits the needs of the system. The development of the migration of Xilema base Web to its version 2.0 was driven by the specifications of the methodology of Agile Unified Process variation of the University of Computer Sciences, obtaining the artifacts generated by its different phases of work. As a solution proposal, two servers were established, one oriented to the view and the templates and the other to the system model. The Angular framework and the Bootstrap development environment for the server in charge of the interface were used as new development technologies, as well as Python and Django as technologies for the server in charge of the system logic. Likewise, other tools and technologies were used in the course of the system life cycle such as: Unified Modeling Language, Visual Paradigm, PostgreSQL, PyCharm, Visual Studio Code and Git.

KEYWORDS

Interface, JavaScript, frameworks, migrate, Xilema base Web.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Las Tecnologías de la Información y Comunicación (TIC) son el conjunto de procesos y productos derivados de las nuevas herramientas (hardware y software), soportes de la información y canales de comunicación relacionados con el almacenamiento, procesamiento y transmisión digitalizada de la información. Agrupan un conjunto de sistemas necesarios para administrar la información, permiten el fácil acceso a una inmensa fuente de información, proporcionan un proceso rápido y fiable de todo tipo de datos, canales de comunicación inmediata, capacidad de almacenamiento, automatización de trabajos, interactividad y la digitalización de toda la información. La revolución tecnológica que vive la humanidad actualmente es debida en buena parte a los avances significativos en las TIC. El mantener un ritmo adecuado de cambio respecto a los avances tecnológicos actuales y establecer mecanismos en la incorporación de las TIC, debe constituir un eje transversal, motivo por el cual debe asumir los retos de la actualidad y establecer acciones de mejoramiento continuo ante la inminente necesidad de hacer frente a todos los desafíos impuestos por la sociedad.(Díaz Lazo, Pérez Gutiérrez, Florido Bacallao 2011)

Dentro del sector de las tecnologías de la información hay diversos roles relacionados con la arquitectura, pero básicamente se pueden hacer varias divisiones: sistemas, datos, almacenamiento, redes y software. Los arquitectos de software; típicamente conocidos como ingenieros de software, diseñan y construyen las aplicaciones que van a permitir ofrecer el servicio que necesitan los usuarios de los sistemas de información u otras aplicaciones. Como un subgrupo de los arquitectos de software se encuentran los arquitectos Web, especializados en diseñar y construir aplicaciones que se van a utilizar a través de lo que se conoce como la Web.(Azaña 2014)

La Web se puede considerar como una plataforma o “sistema operativo” en el cual los recursos están distribuidos en la red y están siendo extendidos en todo momento con posibilidades ilimitadas. Para acceder a un software Web solo se necesita disponer de un navegador de páginas Web. Debido a la arquitectura de las aplicaciones Web, el navegador suele quedar relegado a mostrar la interfaz de usuario, mientras que toda la compleja lógica de negocio se lleva en el lado del servidor. Una aplicación Web es proporcionada por un servidor Web y utilizada por usuarios que se conectan desde cualquier punto, vía clientes Web. La arquitectura de un sitio Web tiene tres componentes principales: un servidor Web, una conexión de red y uno o más clientes (Instituto Tecnológico de Matehuala 2015). Las aplicaciones Web son elaboradas por especialistas en desarrollo Web, antiguamente se realizaban de forma tediosa ya que los desarrolladores debían tener un vasto conocimiento sobre todo el código de la aplicación. Con el propósito de darle solución

a este problema comenzaron a surgir un conjunto de herramientas denominadas marcos de trabajo, por los que en la actualidad es más sencillo desarrollar aplicaciones Web con mayor calidad.

Los marcos de trabajo ofrecen un conjunto de clases e interfaces interrelacionadas en forma de un diseño reutilizable para una línea de productos, con una fuerte cohesión estructural en cuanto a jerarquía de clases, herencia y composición, así como de comportamiento en cuanto a modelo de interacción de los objetos. Un marco de trabajo define un modelo de proceso y de datos común a todas las aplicaciones informáticas de un mismo dominio. No se trata de una aplicación completa, sino que deja un marco que tiene que ser rellenado por el desarrollador del sistema concreto.(Ibáñez, Ballesteros, Gervás 2004)

El desarrollo de la ciencia y la tecnología en la sociedad cubana ha repercutido en el desarrollo económico, político y social del país, evidenciado en la informatización de procesos de la vida diaria, además de la prestación de servicios mediante recursos informáticos. En Cuba, la Universidad de Ciencias Informáticas (UCI) es un pilar de esta rama. Es un centro docente - productor donde se desarrollan aplicaciones y servicios informáticos orientados a diversos sectores de la economía y los servicios, dentro y fuera de país. Presenta un catálogo comercial orientado a cinco líneas de alto impacto, con productos desarrollados sobre plataformas de software libre.(*Productos* | *Universidad de las Ciencias Informáticas* 2018)

Las cinco líneas de alto impacto de la universidad están conformadas por: Xedro como marca para los productos de empresas e instituciones, Xabal como marca para los productos de administración pública, Xauce como marca para los productos de educación, Xavia como marca para los productos de salud y Xilema como marca para los productos de telemática (*Productos* | *Universidad de las Ciencias Informáticas* 2018). La institución cuenta con diversos centros de desarrollo. En el Centro de Telemática (TLM), orientado a las telecomunicaciones y la seguridad informática, se desarrollan diversos sistemas y servicios informáticos, entre los que se destaca el Gestor de Recursos de Hardware y Software (XilemaGRHS), el cual se encarga de realizar inventarios de componentes de hardware y software en una red de computadoras.(*Centro de Telemática (TLM)* | *Universidad de las Ciencias Informáticas* 2018)

XilemaGRHS como arquitectura base utiliza Xilema base Web 1.0, el marco de trabajo que implementa la estrategia marcaria Xilema, el cual tiene como propósito contribuir al desarrollo de nuevas aplicaciones Web logrando una homogeneidad entre todos los productos del Centro de TLM o cualquier otro centro que utilice como tecnologías informáticas de desarrollo Python y Django. Es una arquitectura definida como modular, ya que presenta un área determinada en la cual se pueden agregar o eliminar módulos al sistema que lo utilice según su necesidad, lo cual es ventajoso. También está determinada como Modelo Vista Plantilla

(MTV por sus siglas en inglés), definida como una aplicación de una sola página, la cual cuando se actualiza el sistema carga completa, disminuyendo así el rendimiento y la rapidez.

Xilema base Web es la base que se entrega a los estudiantes a partir de tercer año de la carrera, cuando se incorporan directamente al centro TLM, para la realización de su práctica profesional y así en el transcurso de la misma hasta llegar al trabajo de diploma en quinto año. Actualmente no existe suficiente documentación asociada a Xilema base Web, ninguna específicamente relacionada con la configuración de la interfaz, lo que dificulta en gran medida a cualquier programador que desee relacionarse con el producto, como, por ejemplo, cuando se desea agregar nuevos componentes. Por lo que se hace preciso una capacitación previa al trabajar con el sistema. En la última actualización de Xilema base Web 1.0 se encontraron un grupo de no conformidades, fundamentalmente de usabilidad, en el proceso de calidad, ya que en las pruebas del grupo de calidad se detectaron elementos que deben ser mejorados, como la interfaz de usuario. Por tanto, se hace necesario hacer una evolución del sistema, modificando la interfaz hacia una nueva tecnología donde se logre un software más amigable con el cliente, logrando sostenibilidad en el tiempo.

Xilema base Web fue creado utilizando como tecnología Bootstrap 2.0, versión que utiliza como preprocesador de Hojas de Estilo en Cascada (CSS por sus siglas en inglés) *Less*, el cual al compilar lo hace de una forma lenta y ha dejado de ser popular entre los desarrolladores Web. Actualmente Bootstrap ha avanzado hacia nuevas versiones más adecuadas a las nuevas necesidades de los desarrolladores. El diseño de Xilema base Web 1.0 fue establecido tomando como base la tecnología Backbone.js en su versión 1.1. Como consecuencia, no presenta un diseño adaptativo, el cual permite adaptar el sistema al entorno de navegación del usuario, dígase la multiplicidad de dispositivos a través de los cuales el cliente pueda acceder al sistema, lo que impide su avance hacia la tecnología de dispositivos móviles, siendo solo posible utilizarlo en los navegadores de las computadoras. Asimismo, dicha tecnología no es migrable, lo que frena el progreso de la interfaz del sistema a las nuevas tecnologías que van surgiendo en el desarrollo Web. Por consecuente, Xilema base Web, contiene un conjunto de elementos actuales tales como usabilidad, rendimiento y diseño, los cuales le restan calidad al producto. Por lo que es necesario escalar el sistema con tecnologías actuales, obtenido mayores potencialidades, siendo más eficiente y teniendo mayor rapidez.

En el contexto actual donde los marcos de trabajo JavaScript y el desarrollo de interfaces ha alcanzado altos niveles de perfeccionamiento; es necesaria la utilización de marcos de trabajo estables los cuales brinden una arquitectura sólida y estándar para los proyectos, proporcionando una mejora en la calidad del

software; capaces de originar productos de código abierto, los cuales obtengan gran demanda en el mercado laboral. Por tanto, Backbone.js 1.1, como marco de trabajo queda obsoleto, imposibilitando el avance del desarrollo de Xilema base Web hacia la nueva Web y sus novedosas aplicaciones, así como el uso de nuevos componentes que agilizarían en gran medida los procesos de desarrollo y actuarían de manera positiva en el producto de cara al cliente.

Como resultado del análisis de la situación planteada se identifica el siguiente **problema de investigación:** ¿Cómo mejorar la usabilidad, eficiencia y restricciones de diseño en la interfaz de usuario de Xilema base Web 1.0?

Se tiene como **objeto de estudio:** marcos de trabajo que utilicen JavaScript.

Para dar solución al problema de investigación identificado anteriormente, se define como **objetivo general:** migrar la interfaz de usuario de Xilema base Web 1.0 para mejorar la usabilidad, eficiencia y restricciones de diseño.

Este objetivo estará enmarcado en el **campo de acción:** marco de trabajo de JavaScript para Xilema base Web 1.0.

Para desarrollar satisfactoriamente la investigación se definen un conjunto de **tareas de la investigación** que permiten dar solución al objetivo propuesto:

1. Análisis de las tecnologías homólogas utilizadas en el diseño Web para la posterior migración del software.
2. Selección de las tecnologías, herramientas y metodología a emplear para desarrollar la migración del software.
3. Estudio de las deficiencias de la arquitectura actual del sistema para diseñar e implementar las funcionalidades del sistema.
4. Validación del sistema para garantizar la calidad del software y que cumpla con las necesidades del cliente.

Durante la investigación y desarrollo del presente trabajo de diploma se emplean distintos **métodos científicos,** tanto teóricos como empíricos.

Métodos teóricos:

- Analítico-Sintético: se utiliza en el proceso de análisis y revisión de artículos y documentos relacionados con el tema, de donde se intenta extraer los elementos esenciales que contribuyen a profundizar más sobre la temática, y acoplar de manera íntegra las distintas ideas generadas.

- Inductivo-Deductivo: se utiliza en el proceso del estudio de marcos de trabajo que utilicen JavaScript de forma aislada con el fin de arribar a conclusiones generales, de las cuales luego se llegarán a conclusiones particulares que solucionen el problema a resolver.

- Modelación: se utiliza en el proceso de diseño del objetivo, en el cual se elaboran los modelos y diagramas referentes al problema.

Métodos empíricos

- Observación: se evalúa el funcionamiento y avance de la propuesta de solución, determinando el nivel de satisfacción del cliente.

- Entrevista: se realiza una conversación planificada con el cliente, en la cual se determinan las necesidades del producto.

- Encuesta: se evalúa el nivel de satisfacción del cliente sobre la propuesta de solución.

El presente trabajo está estructurado por 3 capítulos fundamentales, que incluye desde la fundamentación teórica hasta las pruebas de validación realizadas al producto. A continuación, se muestra la estructuración y descripción de los capítulos:

Capítulo 1: se expone la fundamentación teórica donde se realiza el estado del arte, en el cual se aborda el estudio de los marcos de trabajo que utilicen JavaScript actuales que se pueden utilizar para la migración del sistema a las nuevas tecnologías. También se realiza una descripción de las principales herramientas y tecnologías a utilizar en el desarrollo de la investigación, así como sobre la metodología de desarrollo que guía el proceso de creación del software.

Capítulo 2: se presentan las características y el diseño del sistema, por lo que se describe la propuesta de solución a desarrollar, mediante la exposición de sus características principales, realizando las descripciones de requisitos por procesos referentes a las funcionalidades que presenta la solución. Se elaboran los artefactos propuestos por la metodología AUP-UCI.

Capítulo 3: se aborda lo referente a las fases de implementación y pruebas. Se realiza la implementación de los requisitos del sistema, asimismo, las pruebas precisadas por la metodología a utilizar para demostrar que el sistema cumple con las necesidades generadas por el cliente. También se aplican pruebas de aceptación con el fin de verificar la satisfacción del cliente con el funcionamiento del software.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

Para la creación de un sistema informático es de vital importancia que el equipo de desarrollo realice un estudio de las herramientas, tecnologías y metodología más propicias a usarse de acuerdo a las tendencias actuales, esto contribuye a que no se utilicen tecnologías y herramientas obsoletas, ni la metodología inadecuada, lo que puede perjudicar la calidad del proceso productivo y con esto afectar el curso final del sistema en desarrollo. En este capítulo se realiza un estudio de las características de los diferentes marcos de trabajo que utilizan JavaScript existentes en el mundo, para hacer una comparación entre ellos y lograr una solución adecuada para el problema. Se detallarán las características principales de las herramientas y tecnologías informáticas, así como la metodología de desarrollo a utilizar durante el desarrollo del software.

1.1. CONCEPTOS ASOCIADOS

Durante la investigación son analizados una serie de conceptos. En vista de obtener un mejor entendimiento de la solución, se decide realizar una breve explicación de dichos conceptos según diferentes opiniones, donde se llegará a una conclusión particular de cada uno. A continuación, se detallan los principales conceptos analizados.

1.1.1. USABILIDAD

La usabilidad no solo se concibe como facilidad de uso, sino que también implica eficacia en términos de medidas de rendimiento. Por lo tanto, la definición formal propuesta para la usabilidad de un sistema o equipo es: la capacidad en términos funcionales para ser utilizada fácil y efectivamente por el rango especificado de usuarios, dada la capacitación específica y el soporte del usuario, para cumplir con el rango especificado de tareas, dentro del rango especificado de escenarios ambientales.(Shackel, Richardson 1991)

La usabilidad hace referencia a la rapidez y facilidad con que las personas llevan a cabo sus tareas propias a través del uso del producto objeto de interés, idea que descansa en tres puntos: aproximación al usuario, amplio conocimiento del contexto de uso y el producto debe satisfacer las necesidades del usuario. Usabilidad se refiere a la efectividad, eficiencia y satisfacción con la cual usuarios específicos pueden alcanzar metas específicas en ambientes particulares.(Neri 2007)

Como conclusión, se puede considerar usabilidad como la operabilidad y comprensibilidad de una aplicación, con el fin de lograr que el sistema sea capaz de realizar todas las operaciones solicitadas y sea entendible fácilmente por el cliente.

1.1.2. EFICIENCIA

Capacidad de disponer de alguien o de algo para conseguir un efecto determinado (RAE- ASALE 2018a). En términos de informática, la eficiencia es la relación entre el trabajo útil y los recursos gastados. En otras palabras, la relación entre la salida y la entrada de un sistema dado. Un algoritmo es eficiente, si hace un buen trabajo de economizar en los recursos de la computadora para lograr su objetivo. La eficiencia se mide como los recursos gastados por el usuario en relación con la precisión y la integridad de los objetivos logrados. Se logra una alta eficiencia cuando el usuario alcanza sus objetivos mientras gasta la menor cantidad posible de recursos.(Internation Design Foundation 2018)

Como conclusión, se puede considerar eficiencia como el comportamiento de una aplicación en el tiempo, con el fin de lograr que el sistema ejecute correctamente las peticiones solicitadas por el usuario en el menor tiempo posible.

1.1.3. RESTRICCIONES DE DISEÑO

La palabra restricción es el resultado de la acción y efecto de restringir. Limitación o reducción impuesta (RAE- ASALE 2018b). Restricción es una condición o medida límite. La limitación o control de alguien o algo, o el estado de ser restringido.(Oxford Dictionaries 2018)

Diseño se refiere a un boceto, bosquejo o esquema que se realiza (RAE- ASALE 2018c). Es un plan producido para mostrar el aspecto y la función o el funcionamiento de un objeto antes de que se haga. La acción de concebir y producir un plan antes de que se haga. La disposición de las características de un artefacto, como se produce siguiendo un plan. Un patrón decorativo. Propósito o planificación que existe detrás de una acción, hecho u objeto.(Oxford Dictionaries Definition 2018)

Como conclusión, se puede considerar restricciones de diseño como las restricciones que han sido definidas por el cliente para el diseño de un sistema a las cuales es necesario adherirse para el desarrollo del mismo.

1.1.4. JAVASCRIPT

JavaScript es un lenguaje de programación, más utilizado en Internet, para añadir interactividad a las páginas Web. Es un lenguaje interpretado (Vilá 2001). JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas Web dinámicas. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos en JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.(Pérez 2009a)

Como conclusión, se puede considerar JavaScript como el lenguaje de programación que se utilizará para realizar las plantillas de la aplicación web que dará solución al objetivo propuesto.

1.1.5. MARCOS DE TRABAJO

Los marcos de trabajo son instructivos, son guías más claras para las estructuras de resolución de problemas, lo que compensan con facilidad de uso. Usando un marco de trabajo, la construcción de aplicaciones se realiza mucho más rápido (Sanders 2013). Un marco de trabajo es un conjunto de herramientas, librerías, convenciones y buenas prácticas que pretenden encapsular las tareas repetitivas en módulos genéricos fácilmente reutilizable. Los marcos de trabajo se utilizan en el ámbito de la programación de aplicaciones web y recientemente se han comenzado a utilizar para CSS. Los marcos de trabajo más complejos incluyen utilidades para que el programador no tenga que trabajar en ningún aspecto genérico del diseño web.(Pérez 2009b)

Como conclusión, se puede considerar marco de trabajo como una estructura que indica cómo deben construirse una aplicación, proporcionando reglas y pautas que deben ser cumplidas. Es un esquema o esqueleto de elementos interconectados que admite un enfoque particular para un objetivo específico. Entre los marcos de trabajo que existen se encuentran los que utilizan JavaScript. Generalmente se utilizan del lado del cliente y son un conjunto de componentes y bibliotecas que tienen el fin de satisfacer las necesidades de una aplicación web, frecuentemente de una sola página, en navegadores. Entre los marcos de trabajo de JavaScript se encuentran: Angular, ReactJS, VueJS, EmberJS, AureliaJS, MeteorJS, BackboneJS, PolymerJS, KnockoutJS, MercuryJS, entre otros.

1.1.6. AJAX

El término Ajax es un acrónimo de *Asynchronous JavaScript* + Lenguaje de Marcado Extensible (XML por sus siglas en inglés), que se puede traducir como "JavaScript asíncrono + XML" (Pérez 2008). Ajax no es una tecnología. En realidad, se trata de varias tecnologías, cada una progresando por sí misma, uniéndose en nuevas y poderosas formas. Ajax incorpora: presentación basada en estándares usando Lenguaje de Marcado de Hipertexto eXtensible (XHTML por sus siglas en inglés) y CSS, visualización dinámica e interacción utilizando el Modelo de Objetos de Documento, intercambio de datos y manipulación usando XML y las Transformaciones Extensibles del Lenguaje de Hojas de estilo (XSLT por sus siglas en inglés), recuperación asincrónica de datos usando XML Protocolo de Trasferencia de Hipertexto (Http por sus siglas en inglés) *Request* y JavaScript vinculando todo junto.(Garrett 2015)

Como conclusión, se puede considerar Ajax como nuevo motor de JavaScript que se utilizará como parte del marco de trabajo de JavaScript que se seleccionará para la solución del objetivo propuesto.

1.2. ESTUDIO DE LOS MARCOS DE TRABAJO QUE UTILIZAN JAVASCRIPT

Para fundamentar la investigación acerca de los marcos de trabajo que utilizan JavaScript, se decide realizar un estudio sobre los marcos de trabajo más actuales utilizados en el mundo, con el fin de seleccionar el más adecuado, que cumpla con las características necesarias para el desarrollo del sistema.

1.2.1. ANGULAR

Angular no es una biblioteca sino un marco de JavaScript que abarca la extensión del Lenguaje de Marcado de Hipertexto (HTML por sus siglas en inglés) a un formato más expresivo y legible. Permite decorar el HTML con un marcado especial que se sincroniza con el JavaScript y permite escribir la lógica de la aplicación en lugar de actualizar manualmente las vistas. Favorece la escritura de un código más limpio y eficiente. Brinda un fácil desarrollo de aplicaciones Ajax. Es uno de los marcos de trabajo principales de interfaces que utilizan objetos JavaScript simples para la capa del modelo. Esto hace que sea extremadamente fácil integrarse con las fuentes de datos existentes y relacionarse con datos básicos.(Nilesh, Priyanka, Deepak 2014)

Entre sus características primordiales se encuentran: vinculación de datos de dos vías, plantillas, patrón de diseño MTV, inyección de dependencia y directivas. La vinculación de datos de dos vías, es probablemente la característica más atrayente Angular. El enlace de datos bidireccional maneja la sincronización entre el

modelo de objeto de documento (DOM por sus siglas en inglés) y el modelo, y viceversa. Las plantillas, amplían el vocabulario HTML para contener instrucciones sobre cómo se debe proyectar el modelo en la vista. Las plantillas HTML son analizadas por el navegador en el DOM. El DOM se convierte en la entrada al compilador Angular. Usar el DOM como entrada, en lugar de cadenas, es la mayor diferencia que tiene con respecto a otros marcos de trabajo JavaScript. Incorpora los principios básicos detrás del patrón de diseño de software Modelo Vista Controlador (MVC por sus siglas en inglés) original en la forma en que desarrolla las aplicaciones Web del lado del cliente. Angular no implementa MVC en el sentido tradicional, sino algo más cercano al MTV. Tiene un subsistema de inyección de dependencia integrado que ayuda al desarrollador al hacer que la aplicación sea más fácil de desarrollar, comprender y probar. Las directivas, son probablemente el aspecto más osado de Angular, ya que se pueden usar para crear etiquetas HTML personalizadas que sirven como artefactos nuevos y personalizados.(Nilesh, Priyanka, Deepak 2014)

Angular presenta una alta comunidad de desarrolladores de alrededor de 278 mil personas (Angular 2018). Es un marco de trabajo de código abierto e implementa como patrón de diseño MTV. Favorece el fácil desarrollo de aplicaciones Ajax y el diseño adaptativo en las mismas. Brinda seguridad y ligereza a sus aplicaciones Web.

1.2.2. REACT.JS

React es un marco de trabajo de JavaScript, creado para resolver los retos involucrados al desarrollar interfaces complejas con un conjunto de datos que cambian con el tiempo. Es un concepto diferente cuando se trata del desarrollo Web en general. Es un cambio de los flujos de trabajo generalmente aceptados y las mejores prácticas, ya que afronta las convenciones que se han convertido en los estándares para las buenas prácticas del marco de trabajo de JavaScript, al introducir muchos nuevos paradigmas y al cambiar el status de lo que se necesita para crear aplicaciones de JavaScript e interfaces de usuarios escalables y mantenibles. React no presenta como patrón de diseño MVC como otros muchos marcos de trabajo, ya que es una representación más simple, siendo solo el componente Vista de este patrón de diseño. Está hecho con componentes declarativos que describen una interfaz. No utiliza ningún enlace de datos observable al crear una aplicación. Asimismo, es fácil de manipular, ya que se pueden tomar los componentes y combinarlos para crear componentes personalizados que funcionen como el desarrollador desee.(Gackenheimer 2015)

Para evitar la complejidad de las interacciones y el procesamiento posterior requerido, React realiza un renderizado completo de la aplicación. Es también declarativo. Mantiene un flujo de trabajo simple. Resuelve

esto dando al desarrollador un DOM virtual para representar en lugar del DOM real. Encuentra la diferencia entre el DOM real y el DOM virtual y lleva a cabo el número mínimo de operaciones DOM requeridas para alcanzar el nuevo estado.(AM, Sonpatki 2016)

React presenta una alta comunidad de desarrolladores de alrededor de 231 mil personas (React 2018). Es un marco de trabajo que no utiliza el patrón de diseño MTV, sino más bien la plantilla del modelo. Al trabajar solo en la plantilla ofrece productos ligeros. Es de código abierto, propone aplicaciones Ajax con diseño adaptativo y seguridad a sus aplicaciones.

1.2.3. VUE.JS

Vue.js es un marco progresivo para construir interfaces de usuario. A diferencia de otros marcos monolíticos, está diseñado desde cero para ser adaptable incrementalmente. Es capaz de impulsar sofisticadas aplicaciones de una sola página cuando se utiliza en combinación con herramientas modernas y bibliotecas de soporte.(*Introduction — Vue.js* 2018)

Vue.js es un marco de JavaScript moderno para construir una aplicación Web moderna. Es lo suficientemente liviano como para ser utilizado en un sitio Web existente para introducir alguna funcionalidad adicional, o como el marco frontal principal para una aplicación de una sola página (SPA) de grado profesional y de gran escala. Ofrece enlace a datos en dos direcciones, posibilidad de renderizado en el lado del servidor utiliza un DOM virtual.(Kurniawan 2017)

Vue.js presenta una comunidad media de desarrolladores de alrededor de 74,6 mil personas (Vue.js 2018).Es un marco de trabajo que implementa como patrón de diseño MVC. Es de código abierto, brinda aplicaciones Ajax, ligeras, con diseño adaptativo y seguridad.

1.2.4. EMBER.JS

Ember.js es un marco de JavaScript basado en el patrón de diseño de MVC. Incorpora principios y prácticas de diseño comprobados en el desarrollo moderno de aplicaciones basadas en la Web. Es un marco altamente dogmático y se inspira mucho en la convención de *Ruby on Rails* sobre la filosofía de configuración. Convención sobre configuración, también conocida como codificación por convención, es una filosofía de diseño en la que las cosas funcionan como se espera, siempre que sigan un conjunto común de directrices. Ember.js tiene muy bien definidos roles y responsabilidades para cada componente.(Puri 2015)

Es un marco de trabajo para crear aplicaciones Web ambiciosas. Se basa en jQuery, el marco de trabajo de JavaScript que suaviza las incoherencias del navegador y agrega una gran cantidad de funciones de utilidad a JavaScript. Más allá de la tecnología involucrada, es un conjunto de convenciones para construir software robusto y comprobable. Entre sus principales características están: enlace de datos fácil, rápido y bidireccional, ergonomía del desarrollador, Ember Data, que proporciona muchas características de mapeado relacional de objetos (ORM) y se adapta y extrae casi cualquier *back-end,* administración de historial y vistas construidas en HTML.(Cravens, Brady 2014)

Ember.js presenta una comunidad media de desarrolladores de alrededor de 38,9 mil personas (EmberJS 2018).Es un marco de trabajo bastante robusto en sus aplicaciones, utiliza como patrón de diseño MVC. Ofrece seguridad, aplicaciones Ajax de código abierto y con diseño adaptativo.

1.2.5. AURELIA

La experiencia del desarrollador es una fortaleza clave de Aurelia. Depende mucho de las convenciones. La mayoría de esas convenciones son configurables y se pueden cambiar si no se ajustan a sus necesidades. Incorpora los principios de patrón de diseño modelo – vista – vista modelo (MVVM). Los componentes son de gran importancia, ya que es la unión entre una plantilla HTML, llamada vista, y una clase de JavaScript, llamada modelo de vista. La vista es responsable de mostrar el componente, mientras que el modelo de vista controla sus datos y comportamiento. Los componentes pueden usar otros componentes; se pueden componer para formar componentes más grandes o más complejos. Aurelia no es su marco monolítico promedio. Es un conjunto de bibliotecas débilmente acopladas con abstracciones bien definidas. Cada una de sus bibliotecas centrales resuelve un problema específico y bien definido común a las aplicaciones de una sola página. Las bibliotecas centrales se pueden dividir en múltiples categorías. A pesar de que una aplicación Aurelia puede compilarse utilizando cualquier administrador de paquetes, sistema de compilación o *bundler* que desee, la herramienta preferida para administrar un proyecto es la interfaz de línea de comando (CLI).(Guilbault 2016)

Aurelia presenta una baja comunidad de desarrolladores de alrededor de 3744 personas (Aurelia 2018).Es un marco de trabajo que implementa como patrón de diseño MVVM, brindando ligereza en sus productos. Propone aplicaciones Ajax, de código abierto, con seguridad y diseño adaptativo.

1.3. CONCLUSIONES DEL ESTUDIO REALIZADO

Los marcos de trabajo de JavaScript anteriormente descritos fueron realizados con el fin de facilitar el trabajo a los desarrolladores Web, alcanzando altos niveles de perfeccionamiento. Por lo que se debe tener en cuenta un conjunto de características determinadas, basadas en la necesidad de la solución a desarrollar, antes de escoger el marco de trabajo a utilizar. En el caso de Xilema base Web, se debe utilizar como patrón de diseño MTV, ser una aplicación Ajax, ostentar de un diseño adaptativo, ser de código abierto, brindar mayor seguridad y ser lo más ligero posible. Asimismo, se debe tener en cuenta la popularidad que presenta entre los desarrolladores, ya que las tecnologías varían y van avanzando con bastante rapidez en el tiempo. A continuación, en la figura 1 se muestra una gráfica donde se comparan las tendencias en cuanto a búsquedas realizadas en internet de los marcos de trabajo analizados.

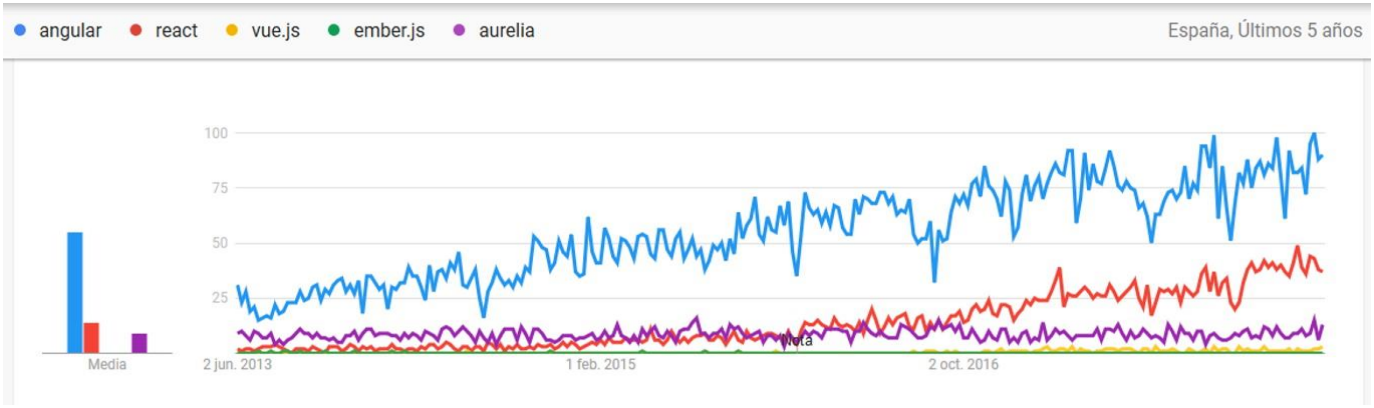


Figura 1: Tendencia de búsqueda de los marcos de trabajo de JavaScript(Google 2018)

A continuación, en la tabla 1 se muestra una comparación entre los marcos de trabajo de JavaScript anteriormente analizados, en cuanto a las características primeramente expuestas necesarias para el desarrollo de la solución e incluyendo el nivel alto, medio o bajo según la cantidad de personas que integran la comunidad de desarrolladores de cada uno, con el fin de obtener una mejor comprensión.

Tabla 1: Comparación entre los marcos de trabajo de JavaScript. (Elaboración propia.)

Marcos de trabajo de JS	MVT	Ajax	Diseño adaptativo	SW ligero	Código abierto	Seguridad	Comunidad de desarrolladores
Angular	✓	✓	✓	✓	✓	✓	Alta(278 mil)
React.js	X	✓	✓	✓	✓	✓	Alta(231 mil)
Ember.js	X	✓	✓	X	✓	✓	Media(38,9 mil)
Vue.js	X	✓	✓	✓	✓	✓	Media(74,6 mil)

Aurelia	X	✓	✓	✓	✓	✓	Baja(3744)

Luego de realizado el análisis anterior se concluye que el marco de trabajo de JavaScript que cumple con todas las características específicas que se necesitan para la migración del software Xilema base Web 1.0; utilizar como patrón de diseño MTV, ser una aplicación Ajax, ostentar de un diseño adaptativo, ser de código abierto, brindar mayor seguridad y ser lo más ligero posible, además que presenta una alta comunidad de desarrolladores; es Angular, por lo que es el seleccionado como marco de trabajo de JavaScript para el desarrollo de la propuesta de solución.

1.4. HERRAMIENTAS Y TECNOLOGÍAS INFORMÁTICAS

1.4.1. ANGULAR 5.2.10

Angular 5.2.10 es un marco de trabajo MTV de JavaScript. Tiene un enfoque novedoso para la creación de plantillas y el enlace de datos bidireccional, lo que hace que el marco sea muy potente y fácil de usar. Con su fuerte énfasis en las pruebas y calidad del código, promueve buenas prácticas para todo el entorno de JavaScript. Dada la calidad y la novedad de la tecnología presenta una creciente popularidad, dado al gran equipo de desarrolladores a cargo de sus mejoras y actualizaciones. Angular se basa en formularios HTML estándar y elementos de entrada. Se encarga de manejar tanto el enlace del modelo del elemento como el enlace del controlador de evento. Brinda un fácil desarrollo de aplicaciones Ajax. Los cambios realizados en el motor de visualización en Angular 4 disminuyen el tamaño del código generado, por tanto, favorecen en cuanto a la disminución del tamaño y la rapidez en las aplicaciones (Darwin, Kozlowski 2013). Se selecciona Angular en su versión 5.2.5, a partir de las ventajas que brinda, descritas anteriormente, como el marco de trabajo de JavaScript que se empleará en la Vista del patrón MTV para garantizar la comunicación entre el cliente y el servidor.

1.4.2. *BOOTSTRAP 4*

Bootstrap 4.0 es un entorno de desarrollo con una serie de recursos que simplifican el desarrollo de un proyecto Web con HTML5, CSS3 y jQuery, de manera que simplifica mucho el trabajo a la hora de diseñar, ya que el marco de trabajo Bootstrap ya tiene una buena parte del trabajo hecho lo cual simplifica mucho la tarea del desarrollo. La rapidez por la cantidad de trabajo que está hecho y muchos componentes que son necesarios están desarrollados previamente. Se integra con HTML5 y CSS3 lo cuál lo hace muy poderoso y por tanto mucho más ligero de cara a los navegadores. Bootstrap es uno de los entornos más utilizados

en la red, ya que está disponible en código abierto (Bhaumik 2015). Se selecciona Bootstrap en su versión 4.0 como el entorno de desarrollo que se empleará en la Plantilla del patrón MTV para obtener una imagen más amigable del producto. Favorece el diseño adaptativo del software, dígase la multiplicidad de dispositivos utilizados como entorno de navegación por el cliente. Es la versión estable más reciente del entorno.

1.4.3. *PYTHON 2.7*

Python 2.7 es un lenguaje de programación interpretado, orientado a objetos y de alto nivel con semántica dinámica. Sus estructuras de datos integradas de alto nivel, combinadas con el tipado dinámico y el enlace dinámico, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como también para usarlo como *scripting* o lenguaje de pegado para conectar componentes existentes. La sintaxis simple y fácil de aprender de Python enfatiza la legibilidad y, por lo tanto, reduce el costo del mantenimiento del programa. Python admite módulos y paquetes, lo que fomenta la modularidad del programa y la reutilización de código. El intérprete de Python y la extensa biblioteca estándar están disponibles en formato fuente o binario sin cargo para todas las plataformas principales, y se pueden distribuir libremente (*What is Python? Executive Summary* 2018). Se selecciona Python en su versión 2.7 como el lenguaje de programación que se empleará en el Modelo del patrón MTV para la configuración en el servidor, a partir de las características anteriormente expuestas.

1.4.4. *DJANGO 1.8*

Django 1.8 como marco de trabajo Web implementado sobre el lenguaje de programación Python, pertenece a la licencia Distribución de software de *Berkeley* (BSD). Django brinda estructura al código fuente, fomentando las buenas prácticas de desarrollo Web, lo que promueve un código legible y fácil de mantener. La implementación del patrón de diseño *Model Template View* (MTV) es una característica propia que contiene el marco de trabajo, la cual contribuye a la organización de las distintas partes de la aplicación, y a modificar estas sin afectar cualquier otra pieza del software. Su alta escalabilidad le posibilita manejar el crecimiento continuo de trabajo de manera fluida sin perder calidad en los servicios (Holovaty, Mosss, M 2015). Se selecciona Django en su versión 1.8 como el marco de trabajo Web, el cual junto al lenguaje de programación Python trabaja desde el Modelo del patrón MTV para la relación almacenamiento-obtención de datos entre el servidor y la base de datos.

1.4.5. *UML 2.0*

El “Lenguaje de Modelado Unificado”, del inglés *Unified Modeling Language* (UML), es un lenguaje basado en diagramas para la especificación, visualización, construcción y documentación de cualquier sistema complejo (Alfonso [no date]). Es la sucesión de una serie de métodos de análisis y diseño orientado a objetos. El lenguaje de modelado es la notación (principalmente gráfica) que usan los métodos para expresar un diseño. El proceso indica los pasos que se deben seguir para llegar a un diseño.(Cornejo 2008)

Por tanto, UML es un lenguaje para describir modelos. Básicamente, un modelo es una simplificación de la realidad que construimos para comprender mejor el sistema que queremos desarrollar. Un modelo proporciona los “planos” de un sistema, incluyendo tanto los que ofrecen una visión global del sistema como los más detallados de alguna de sus partes. UML incorpora toda una serie de diagramas y notaciones gráficas y textuales destinadas a mostrar el sistema desde las diferentes perspectivas, que pueden utilizarse en las diferentes fases del ciclo de desarrollo del software (Alfonso [no date]). Se selecciona ULM en su versión 2.0 como el lenguaje que se utilizará para modelar los diagramas y modelos necesarios en el desarrollo del software, tales como: modelo conceptual, modelo de base de datos, diagrama de paquetes, de clases y de despliegue propios del sistema.

1.4.6. *VISUAL PARADIGM FOR UML 8.0*

Visual Paradigm 8.0 es una herramienta UML. La herramienta está diseñada para una amplia gama de usuarios, incluidos ingenieros de software, analistas de sistemas, analistas de negocios y arquitectos de sistemas, o para cualquier persona interesada en la construcción fiable de sistemas de software de gran escala con un enfoque orientado a objetos. Además, admite los últimos estándares de notación UML. Es un proveedor líder de soluciones de software que permiten a las organizaciones desarrollar aplicaciones de calidad de manera más rápida, mejor y más económica. Visual Paradigm está dedicado a desarrollar y entregar continuamente software, servicios y asociaciones para ayudar a los clientes a transformar con precisión los requisitos de su sistema en soluciones de software de alta calidad, todas con un riesgo mínimo y un máximo retorno de la inversión. Todos los productos de Visual Paradigm están diseñados y desarrollados para eliminar la complejidad, mejorar la productividad y comprimir los plazos de tiempo de desarrollo de software de los clientes (*Visual Paradigm 8.0 (formerly VP-UML 8.0) Released* 2010). Se selecciona Visual Paradigm *for* UML en su versión 8.0 como la herramienta que se utilizará para realizar los diagramas y modelos necesarios en el desarrollo del software definidos en el lenguaje de modelado UML.

1.4.7. *POSTGRESQL 9.4*

PostgreSQL 9.4 es un sistema de administración de bases de datos de propósito general y objeto-relacional, el sistema de base de datos de código abierto más avanzado. Es un software gratuito y de código abierto. Su código fuente está disponible bajo la licencia PostgreSQL, una licencia libre de código abierto (*What is PostgreSQL* 2018). PostgreSQL es el primer sistema de administración de base de datos que implementa la característica de control de concurrencia de múltiples versiones (MVCC), incluso antes de Oracle. Es un sistema de administración de bases de datos relacionales de objetos de propósito general. Permite agregar personalizado a funciones desarrolladas usando diferentes lenguajes de programación como Java. Está diseñado para ser extensible. Se pueden definir personalizaciones en los tipos de datos, tipos de índice, idiomas funcionales (*What is PostgreSQL* 2018). Se selecciona PostgreSQL en su versión 9.4, a partir de las ventajas descritas anteriormente, como el gestor de base de datos que se utilizará para el manejo y obtención de los datos referentes al sistema a desarrollar.

1.4.8. *PYCHARM 2016.3.2*

PyCharm 2016.3.2 es un entorno de desarrollo integrado (IDE) que proporciona la finalización inteligente de códigos, inspecciones de códigos, resaltado de errores sobre la marcha y soluciones rápidas, junto con refactorizaciones automáticas de códigos y capacidades de navegación avanzadas. Brinda soporte de primera clase para Python, JavaScript, TypeScript, CSS. Domina código con reconocimiento de idioma, detección de errores y arreglos de código sobre la marcha. Contiene una búsqueda inteligente para saltar a cualquier clase, archivo o símbolo, o incluso cualquier ventana de herramientas. Las refactorizaciones específicas del lenguaje y del marco ayudan a realizar cambios en todo el proyecto.(*Features - PyCharm* 2018)

La enorme colección de herramientas de PyCharm incluye un depurador y corrector de prueba integrado y herramientas de base de datos incorporadas como PostgreSQL. Contiene un potente depurador con una interfaz gráfica de usuario para Python y JavaScript. Permite crear y ejecutar sus pruebas con asistencia de codificación y un corredor de prueba basado en Interfaz Gráfica del Usuario (GUI) (*Features - PyCharm* 2018). Se selecciona PyCharm en su versión 2016.3.2 como el IDE que permitirá realizar la unificación entre el lenguaje de programación Python y el marco de trabajo Django que se utilizan para la codificación del software en el lado del servidor.

1.4.9. VISUAL STUDIO CODE 1.18.1

Visual Studio Code 1.18.1 es un editor de código fuente ligero pero potente, siendo una aplicación de escritorio que está disponible para Windows, MacOS y Linux. Contiene soporte integrado para *JavaScript, TypeScript y Node.js* y tiene una amplia variedad de extensiones para otros lenguajes como C ++, C #, Java, Python, PHP, Go y tiempos de ejecución como .NET y Unity. Posee características como: terminación de código inteligente, depuración optimizada, edición rápida y potente, navegación de código y refactorización, control de fuente en el producto (*Documentation for Visual Studio Code* 2018). Se selecciona Visual Studio Code en su versión 1.18.1 como el IDE que permitirá realizar la unificación entre el marco de trabajo de JavaScript Angular y el entorno de desarrollo Bootstrap que se utilizan para la codificación del software en el lado del cliente.

1.4.10. GIT 2.13.2

Git 2.13.2 es un sistema de control de versiones distribuidas de código abierto y gratuito diseñado para manejar todo, desde proyectos pequeños a muy grandes, con velocidad y eficiencia. Es fácil de aprender y tiene un rendimiento increíblemente rápido. Supera a las herramientas de Gestión de Configuración de Software (SCM) como *Subversion, CVS, Perforce y ClearCase* con funciones como ramificación local barata, áreas de preparación conveniente y flujos de trabajo múltiples (*Git version control* 2018). Se selecciona Git en su versión 2.13.2, a partir de las ventajas que brinda descritas anteriormente, como el controlador de versiones que se utilizará para el sistema a desarrollar.

1.5. METODOLOGÍA DE DESARROLLO DE SOFTWARE

La metodología es un conjunto de métodos y técnicas de rigor científico que se aplican sistemáticamente durante un proceso de investigación para alcanzar un resultado teóricamente válido. En este sentido, la metodología funciona como el soporte conceptual que rige la manera en que se aplican los procedimientos en una investigación. La metodología de desarrollo del software son métodos que indican cómo hacer más eficiente el desarrollo de sistemas de información. Para ello suelen estructurar en fases la vida de dichos sistemas con el fin de facilitar su planificación, desarrollo y mantenimiento.(John, Julianny, Maria Gabriela, Jeferson 2009)

Las metodologías de desarrollo ágil buscan elaborar software totalmente funcional en el tiempo o plazo establecido para el desarrollo del proyecto. Utilizan un proceso ágil, por tanto, si los requerimientos del

software cambian en cualquier etapa en la que se encuentre el proyecto, el equipo debe adaptar el producto a estos cambios ya que la agilidad es la respuesta efectiva al cambio.(Cevallos 2015)

El Proceso Unificado Ágil (AUP) es una versión simplificada del Proceso Unificado Racional (RUP). Describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. El AUP aplica técnicas ágiles incluyendo desarrollo dirigido por pruebas, modelado ágil, gestión de cambios ágil, y refactorización de base de datos para mejorar la productividad. Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva.(G, arillas 2017)

1.5.1. *AUP-UCI*

AUP-UCI es una variación de la metodología AUP, al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto, exigiéndose así que el proceso sea configurable. Se realizó una variación de la metodología AUP, de forma tal que se adaptara al ciclo de vida definido para la actividad productiva de la UCI.(Sánchez 2014)

Una metodología de desarrollo de software tiene entre sus objetivos aumentar la calidad del software que se produce, de ahí la importancia de aplicar buenas prácticas, para ello se apoyaron en el Modelo de Madurez de la Capacidad Integrado (CMMI-DEV) v1.3. El cual constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora. Estas prácticas se centran en el desarrollo de productos y servicios de calidad.(Sánchez 2014)

De las 4 fases que propone AUP, inicio, elaboración, construcción y transición, se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, a la que se denomina Ejecución y se agrega una fase de Cierre.(Sánchez 2014)

1. Inicio: durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
2. Ejecución: en esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo

se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el producto es transferido al ambiente de los usuarios finales o entregado al cliente. Además, en la transición se capacita a los usuarios finales sobre la utilización del software.

3. Cierre: en esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.(Sánchez 2014)

AUP-UCI presenta diferentes escenarios los cuales se ajustan a las necesidades específicas de cada centro de desarrollo de la UCI, los cuales son: historias de usuario, descripción de requisitos por casos de uso y descripción de requisitos por procesos. En el Centro de TLM, el proyecto de GRHS utiliza el escenario Descripción de requisitos por procesos, por lo que se decide trabajar sobre ese escenario, con el fin de garantizar que la documentación que se genere como parte de la realización del sistema pueda ser reutilizada por el proyecto.

1.6. CONCLUSIONES PARCIALES

- ✓ La investigación acerca de las tecnologías homólogas brindó un mayor entendimiento y conocimiento al desarrollador sobre las tendencias de diseño Web, lo cual permitió la adecuada selección de las tecnologías y herramientas a utilizar en el desarrollo del sistema.

- ✓ La correcta selección de las herramientas y tecnologías de desarrollo garantizó un adecuado desarrollo del software.

- ✓ La selección de la metodología de desarrollo de software favoreció el adecuado diseño del sistema, así como la realización de cada una de las fases del ciclo de vida del software. Mediante la cual, se documentará todo el ciclo de desarrollo de la investigación.

CAPÍTULO 2: CARACTERÍSTICAS Y DISEÑO DEL SISTEMA.

Para desarrollar un sistema informático resulta vital entender la lógica de negocio que plantea el cliente, para luego realizar todas las acciones que en su final le brindarán a este la mejor solución posible. En este capítulo se describe la propuesta de solución a desarrollar, por lo que se exponen sus características principales, realizando los artefactos relacionados con la fase de ejecución definidos por la metodología seleccionada. Asimismo, se efectúan las descripciones de requisitos por procesos referentes a las funcionalidades que presenta la solución, así como se elaboran los casos de prueba con los que se analizará el correcto funcionamiento del sistema.

2.1. PROPUESTA DE SOLUCIÓN

Para darle solución al problema planteado se decide realizar una migración del sistema Xilema base Web, obteniendo la versión 2.0, utilizando el marco de trabajo de JavaScript Angular 5.2.10 y Bootstrap 4 como nuevas tecnologías de desarrollo. Xilema base Web es la arquitectura base que utiliza el Centro de TLM como estrategia marcaria para garantizar la homogeneidad de sus productos. Dichos productos en su mayoría son bastantes amplios, conteniendo grandes volúmenes de información, como es el caso de XilemaGRHS, por lo cual realizar su migración completa tiene una gran demanda de recursos tanto materiales como de fuerza de trabajo y sobre todo del factor tiempo.

Se propone el desarrollo del módulo Sistema, ya que será la base que servirá de guía para completar la migración de los otros productos del centro. Inicialmente el usuario debe ser autenticado por el sistema, luego en dicho módulo se muestra la información referente a la gestión de usuario, la gestión de grupo y los permisos de grupo. La gestión de usuario es un proceso mayor, que trae consigo la adición, modificación y eliminación de información de los usuarios en la base de datos, así como, mostrar los detalles de un usuario, cambiar la contraseña de un usuario, listar y buscar usuarios. La gestión de grupo es un macroproceso que permite adicionar, modificar y eliminar la información de los grupos en la base de datos, asimismo, listar y buscar grupos. La asignación de permisos de grupo es un proceso que admite listar, buscar y guardar permisos, así como, seleccionar grupo.

Se tendrá un servidor utilizando como tecnologías de desarrollo Python 2.7 y Django 1.8, el cual se encargará de la parte lógica del sistema, dígase el modelo del patrón MTV, en el cual se ofrecerán los servicios necesarios a través del protocolo *restful* brindado por la librería *Django-rest-framework*. Se contará con un segundo servidor utilizando como tecnología el marco de trabajo Angular 5.2.10, el cual trabajará

como la vista del patrón MTV, consumiendo los servicios brindados por el otro servidor. Asimismo, interactuará con el cliente a partir de la interfaz de usuario. Se empleará el *Json Web Token* para la comunicación entre los servidores.

La comunicación entre el cliente y el servidor de Angular se establecerá a partir de una solicitud HTTP (http *request*) utilizando un Motor Ajax entre el cliente y el servidor. Desde el momento en que el usuario realiza una solicitud al servidor, comienza la ocurrencia de un conjunto de llamadas, solicitudes y procesamiento de datos en el sistema, con el fin de garantizar una respuesta a dicha solicitud. Primeramente, se realiza una llamada JavaScript a la Vista del patrón MTV implementado en el sistema, donde se encuentra el Motor Ajax, dicha llamada se convierte en una solicitud HTTP que es enviada al servidor. Posteriormente, el servidor realiza el procesamiento de los datos con el gestor de base de datos, obteniendo datos XML que son enviados nuevamente a la Vista, procesados por el Motor Ajax y enviados al cliente en formato HTML + CSS, obteniendo así la respuesta de la solicitud realizada.

Con la propuesta de solución expuesta anteriormente se logra cumplir con el objetivo de migrar la interfaz de usuario de Xilema base Web 1.0 a un nuevo marco de trabajo de JavaScript para mejorar la usabilidad, eficiencia y restricciones de diseño, ya que al utilizar nuevas tecnologías de desarrollo se logra actualizar el motor de JavaScript utilizado para desarrollar la interfaz de usuario. Asimismo, con la utilización de dos servidores que brindan y consumen servicios respectivamente se hace posible modificar cada uno de los servidores de forma independiente sin afectar al otro, incluso se puede cambiar completamente de tecnología. También se pueden generar aplicaciones móviles que consuman los servicios brindados por el servidor de Python y Django. A continuación, en la figura 2 se muestra de forma visual la propuesta de solución anteriormente expuesta.

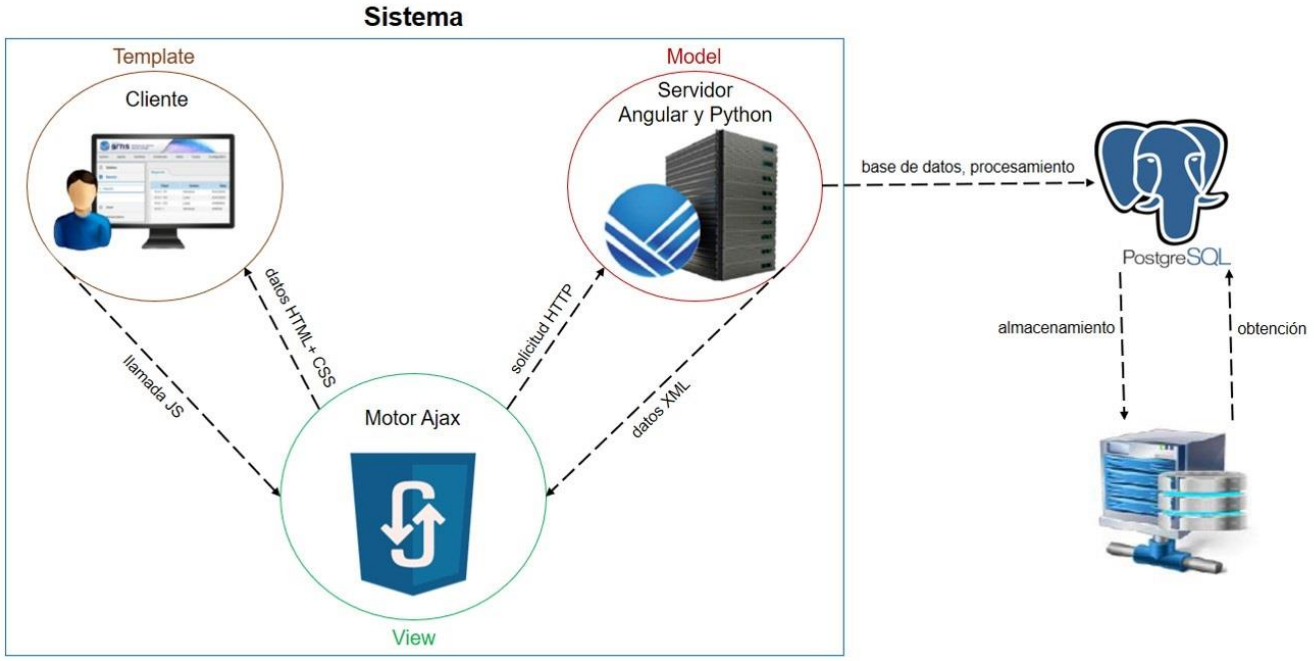


Figura 2: Propuesta de solución de Xilema base Web 2.0

2.2. MODELO CONCEPTUAL

En el modelo conceptual se describe cómo se relacionan los conceptos de un problema. Se utiliza para representar un problema de manera gráfica a través del diagrama de entidad relación (*Data Modeling - UML Diagramming Software* 2010). A continuación, en la figura 3 se muestra el modelo conceptual en el cual se va a describir desde una forma general la relación conceptual entre la UCI y sus centros de desarrollo, hasta llegar al punto particular del software Xilema base Web 2.0.

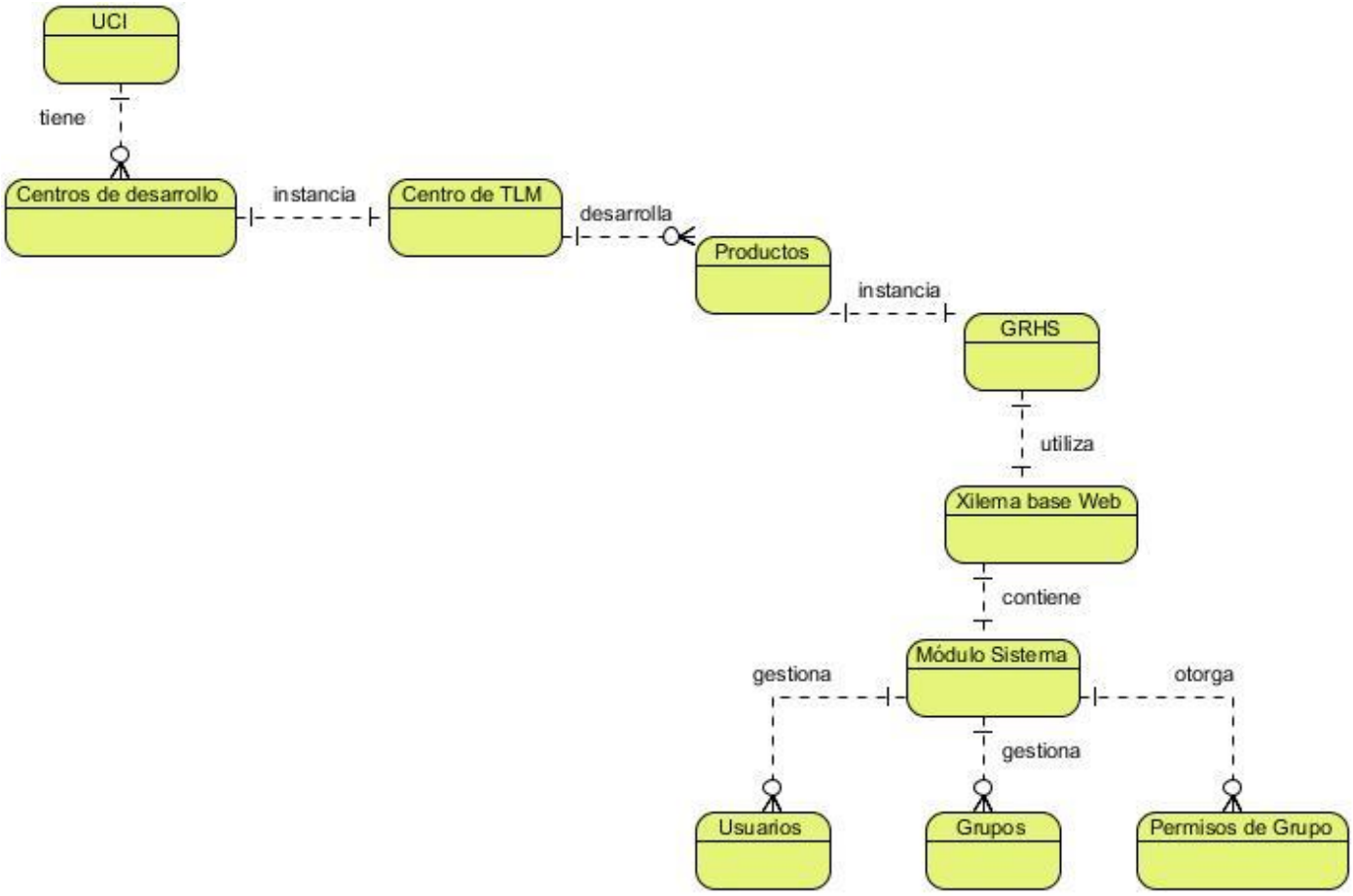


Figura 3: Modelo conceptual de Xilema base Web 2.0

2.2.1. DICCIONARIO DE DATOS

En el diccionario de datos se muestran los datos de cada una de las entidades del modelo conceptual. Se describen cada uno de los atributos teniendo en cuenta como parámetros: nombre, descripción, tipo, si el valor es nulo o único y las restricciones de las clases válidas y no válidas. A continuación, se muestra la descripción de la entidad Usuarios, el resto de las descripciones se encuentra en Anexo1.

2.2.1.1. ENTIDAD USUARIOS

Tabla 2: Descripción de la entidad _ Usuarios

Descripción	*Realiza la gestión de usuarios del sistema.*				
Atributos					
Nombre	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**

					Clases válidas	Clases no válidas
usuario	Nombre del usuario establecido en el sistema.	cadena de caracteres	no	si	Toma valor cadena de caracteres alfanuméricos.	Toma valor caracteres especiales.
contraseña	Contraseña definida por el usuario.	cadena de caracteres y símbolos especiales	no	si	Toma valor cadena de caracteres alfanuméricos y caracteres especiales.	Toma valor no definido como válido.
confirmar contraseña	Confirmación de la contraseña definida por el usuario.	cadena de caracteres y símbolos especiales	no	si	Toma valor cadena de caracteres alfanuméricos y caracteres especiales.	Toma valor no definido como válido.
nombre	Nombre de la persona referente al usuario.	cadena de caracteres	no	si	Toma valor cadena de caracteres.	Toma valor cadena de caracteres alfanuméricos y caracteres especiales.
apellido	Apellido de la persona referente al usuario.	cadena de caracteres	no	si	Toma valor cadena de caracteres.	Toma valor cadena de caracteres alfanuméricos y caracteres especiales.
dirección de correo	Dirección de correo definida por el usuario.	cadena de caracteres y símbolos especiales	no	si	Toma valor cadena de caracteres alfanuméricos y caracteres especiales.	Toma valor no definido como válido.
activo	El usuario puede acceder al sistema o no.	selección	si	si	Toma valor verdadero o falso.	
superusuario	El usuario puede acceder a todos los permisos del sistema o no.	selección	si	si	Toma valor verdadero o falso.	
token	El usuario puede consumir los recursos rest del sistema.	selección	si	si	Toma valor verdadero o falso.	

grupo	Grupo seleccionado para el usuario.	selección múltiple	no	no	Toma valor administrador.	Toma valor no definido como válido.
dominio	Dominio seleccionado para el usuario.	selección múltiple	no	no	Toma valor UCI.CU o local.	Toma valor no definido como válido.

2.3. DESCRIPCIÓN DE REQUISITOS FUNCIONALES

Los requerimientos para un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos requerimientos reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema. Un requerimiento es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste. Es una definición detallada y formal de una función del sistema.(Sommerville 2005)

Requerimientos funcionales: Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer.(Sommerville 2005)

2.3.1. ESPECIFICACIÓN DEL REQUISITO: AUTENTICAR USUARIO

Descripción textual del requisito:

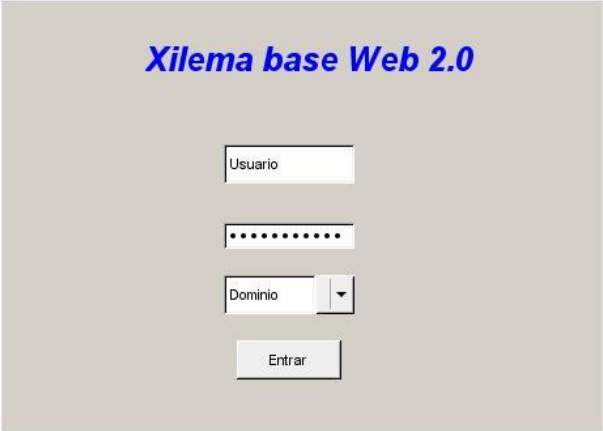
El usuario llena los campos de texto mostrados por el sistema (usuario, contraseña, dominio). El sistema valida los datos con los valores de la base de datos. Si los datos son correctos, el sistema autentica al usuario. Si los datos no son correctos, el sistema deniega al usuario mostrando una notificación donde se explica el motivo por el cual ha sido denegado. Luego el sistema permite al usuario volver a intentar autenticarse con los valores correctos.

Tabla 3: Especificación del requisito Autenticar usuario

Precondiciones	
Flujo de eventos	
Flujo básico Autenticar Usuario	
1.	Llenar el campo de texto nombre de usuario
2.	Llenar el campo de texto contraseña
3.	Seleccionar dominio
4.	Seleccionar botón “Entrar”

Pos-condiciones	
1.	Validar valores con la base de datos
Flujos alternativos	
Flujo alternativo 4.a Campos vacíos	
1.	Mostrar notificación: El nombre de usuario o la contraseña son incorrectos.
2.	Volver al paso 1 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 4.b Valores incorrectos	
1	Mostrar notificación: El nombre de usuario o la contraseña son incorrectos.
2	Volver al paso 1 del flujo básico
Pos-condiciones	
1	N/A
Flujos alternativos	
Flujo alternativo 4.c No se selecciona dominio	
1.	Mostrar notificación: Seleccionar dominio.
2.	Volver al paso 3 del flujo básico
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios

Conceptos	**Usuarios**	Visibles a la interfaz: Usuario, contraseña, dominio.
Requisitos especiales	Funcionabilidad, Fiabilidad.	
Asuntos pendientes		
Prototipo elemental de interfaz gráfica de usuario		



2.3.2. ESPECIFICACIÓN DEL REQUISITO: GESTIONAR USUARIO

Descripción textual del requisito:

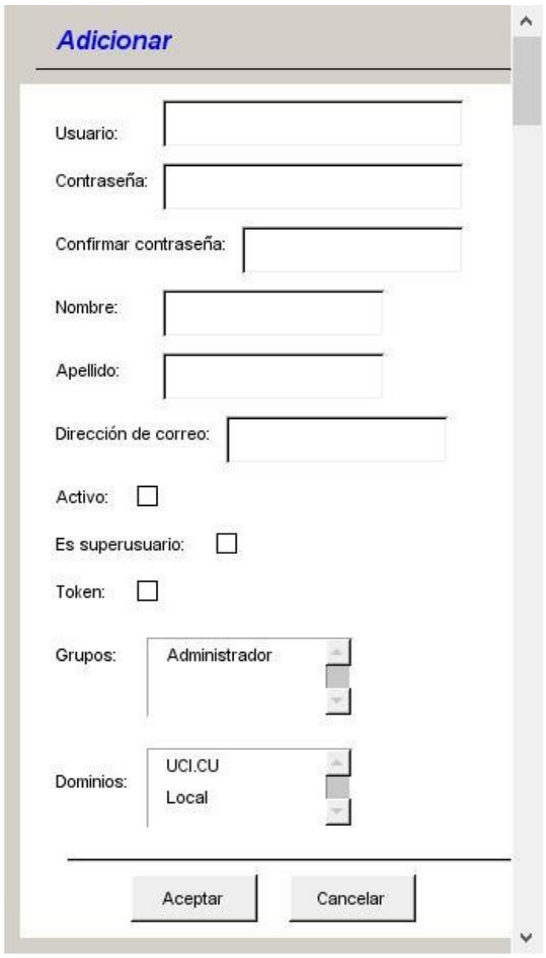
Gestionar usuario es un macroproceso que lleva en sí otros requisitos tales como: adicionar usuario, modificar usuario, eliminar usuario, mostrar detalles de usuario, cambiar contraseña, listar usuarios y buscar usuario. A continuación, se presenta la descripción del requisito Adicionar usuario, el resto de las descripciones referentes a la gestión de usuario se encuentran en Anexo2.

Tabla 4: Especificación del requisito Gestionar usuario -> Adicionar usuario

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar usuario	
Flujo básico Adicionar usuario	
1.	Llenar el campo de texto usuario
2.	Llenar el campo de texto contraseña
3.	Llenar el campo de texto confirmar contraseña
4.	Llenar el campo de texto nombre
5.	Llenar el campo de texto apellido
6.	Llenar el campo de texto dirección de correo
7.	Seleccionar el campo activo
8.	Seleccionar el campo superusuario
9.	Seleccionar el campo token
10.	Llenar el campo de texto grupo
11.	Llenar el campo de texto dominio
12.	Seleccionar botón “Aceptar”
13.	Enviar valores
Pos-condiciones	
1.	Adicionar valores en la base de datos
Flujos alternativos	
Flujo alternativo 12.a Campos vacíos	
1.	Señalar campos vacíos

2.	Mostrar notificación: Por favor inserte un valor para este campo.
3.	Llenar campos de texto vacíos
4.	Volver al paso 12 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 12.b Valores incorrectos	
1.	Señalar campos con valores incorrectos
2.	Mostrar notificación referente a las clases válidas para cada campo
3.	Llenar los campos de textos incorrectos
4.	Volver al paso 12 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 12.c Usuario existente	
1.	Señalar el campo usuario
2.	Mostrar notificación: El nombre de usuario ya existe.
3.	Actualizar el campo de texto usuario
4.	Volver al paso 12 del flujo básico
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios

Conceptos	**Usuarios**	Visibles a la interfaz: Usuario, contraseña, confirmar contraseña, nombre, apellidos, dirección de correo, activo, superusuario, token, grupo, dominio.
Requisitos especiales	Usabilidad, Operabilidad.	
Asuntos pendientes		
Prototipo elemental de interfaz gráfica de usuario		



2.3.3. *ESPECIFICACIÓN DEL REQUISITO: GESTIONAR GRUPO*

Descripción textual del requisito:

Gestionar grupo es un macroproceso que lleva en sí otros requisitos tales como: adicionar grupo, modificar grupo, eliminar grupo, listar grupos y buscar grupo. A continuación, se presenta la descripción del requisito Adicionar grupo, el resto de las descripciones referentes a la gestión de grupo se encuentran en Anexo3.

Tabla 5: Especificación del requisito Gestionar grupo -> Adicionar grupo

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar grupo	
Flujo básico Adicionar grupo	
1.	Llenar el campo de texto nombre
2.	Seleccionar botón “Aceptar”
3.	Enviar valores
Pos-condiciones	

1.	Adicionar valores en la base de datos
Flujos alternativos	
Flujo alternativo 2.a Campos vacíos	
1.	Señalar campos vacíos
2.	Mostrar notificación
3.	Llenar campos de texto vacíos
4.	Volver al paso 2 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 2.b Valores incorrectos	
1.	Señalar campos con valores incorrectos
2.	Mostrar notificación referente a las clases válidas para cada campo
3.	Llenar los campos de textos incorrectos
4.	Volver al paso 2 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 2.c Grupo existente	
1.	Señalar el campo nombre
2.	Mostrar notificación: El nombre ya existe.
3.	Actualizar el campo de texto nombre
4.	Volver al paso 2 del flujo básico
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 21: Descripción de la entidad _ Grupos.

Conceptos	**Grupos**	Visibles a la interfaz: Nombre
Requisitos especiales	Usabilidad, Operabilidad.	
Asuntos pendientes		

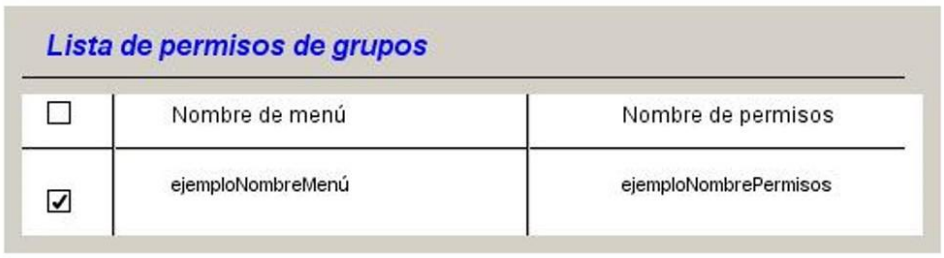
Prototipo elemental de interfaz gráfica de usuario

2.3.4. ESPECIFICACIÓN DEL REQUISITO: PERMISOS DE LOS GRUPOS

Descripción textual del requisito:

Permisos de los grupos es un macroproceso que lleva en sí otros requisitos tales como: listar permisos, buscar permisos, seleccionar grupo y guardar permisos. A continuación, se presenta la descripción del requisito Listar permisos, el resto de las descripciones referentes a los permisos de los grupos se encuentran en Anexo4.

Tabla 6: Especificación del requisito Permisos de los grupos -> Listar permisos

<table>
<tr><td colspan="2">Precondiciones</td><td colspan="2">El cliente ha sido validado.</td></tr>
<tr><td colspan="4">Flujo de eventos Permisos de los grupos</td></tr>
<tr><td colspan="4">Flujo básico Listar permisos</td></tr>
<tr><td>1.</td><td colspan="3">Listar los permisos existentes en la base de datos</td></tr>
<tr><td>2.</td><td colspan="3">Mostrar de todos los permisos el valor nombre de menú</td></tr>
<tr><td>3.</td><td colspan="3">Mostrar de todos los permisos el valor nombre de permiso</td></tr>
<tr><td colspan="4">Pos-condiciones</td></tr>
<tr><td>1.</td><td colspan="3">N/A</td></tr>
<tr><td colspan="4">Validaciones</td></tr>
<tr><td>1.</td><td colspan="3">Ver Figura 3: Modelo conceptual de Xilema base Web 2.0 y Tabla 22: Descripción de la entidad _ Permisos de los grupos.</td></tr>
<tr><td colspan="2">Conceptos</td><td>Permisos de Grupos</td><td>Visibles a la interfaz:
Nombre de menú, nombre de permiso</td></tr>
<tr><td colspan="2">Requisitos especiales</td><td colspan="2">Funcionabilidad, Precisión.</td></tr>
<tr><td colspan="2">Asuntos pendientes</td><td colspan="2"></td></tr>
<tr><td colspan="4">Prototipo elemental de interfaz gráfica de usuario</td></tr>
<tr><td colspan="4"> 
</td></tr>
</table>

2.4. DESCRIPCIÓN DE REQUISITOS NO FUNCIONALES

Requerimientos no funcionales: Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no

funcionales a menudo se aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema.(Sommerville 2005)

2.4.1. ESPECIFICACIÓN DEL REQUISITO NO FUNCIONAL: USABILIDAD

Tabla 7: Especificación del requisito no funcional Usabilidad -> Comprensibilidad

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	Comprensibilidad
Objetivo	Lograr que el sistema sea entendible fácilmente para los usuarios.
Origen	El usuario
Artefacto	El sistema
Entorno	El sistema está funcionando correctamente.
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	
NA	

Tabla 8: Especificación del requisito no funcional Usabilidad -> Operabilidad

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	Operabilidad
Objetivo	Lograr que el sistema sea capaz de realizar todas las operaciones solicitadas por el cliente.
Origen	El usuario
Artefacto	El sistema
Entorno	El sistema está funcionando correctamente.
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	
NA	

2.4.2. ESPECIFICACIÓN DEL REQUISITO NO FUNCIONAL: FUNCIONABILIDAD

Tabla 9: Especificación del requisito no funcional Funcionabilidad -> Precisión

Atributo de Calidad	Funcionabilidad
Sub-atributos/Sub-características	Precisión
Objetivo	Verificar que el sistema arroje resultados o efectos acordes a las necesidades para las cuales fue creado.
Origen	Cliente del producto final
Artefacto	El sistema

Entorno	El sistema está funcionando correctamente
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	
NA	

Tabla 10: Especificación del requisito no funcional Funcionabilidad -> Fiabilidad

Atributo de Calidad	Funcionabilidad
Sub-atributos/Sub-características	Fiabilidad
Objetivo	Prevenir el acceso no autorizado, ya sea accidental o premeditado, a los datos.
Origen	El usuario
Artefacto	Canal de Comunicación, Sistema
Entorno	El usuario desea obtener información sobre los datos procesados por el sistema.
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a Persona no autorizada intenta acceder a los datos del sistema.	
	Bloquear el acceso al sistema.
Medida de respuesta	
1 segundo	
2.a Persona con permisos restringidos intenta acceder a información a la que no tiene acceso.	
	Bloquear el acceso a estos datos.
Medida de respuesta	
1 segundo	

2.4.3. ESPECIFICACIÓN DEL REQUISITO NO FUNCIONAL: EFICIENCIA

Tabla 11: Especificación del requisito no funcional Eficiencia -> Utilización de recursos

Atributo de Calidad	Eficiencia
Sub-atributos/Sub-características	Utilización de recursos
Objetivo	La cantidad mínima de RAM es 256 MG. La capacidad mínima de disco duro a utilizar el servidor y servidor de base de datos es 20GB.
Origen	Externo al sistema
Artefacto	Hardware del cliente y los servidores
Entorno	Operación normal / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA
Medida de respuesta	

NA

Tabla 12: Especificación del requisito no funcional Eficiencia ->Comportamiento en el tiempo

Atributo de Calidad	Eficiencia
Sub-atributos/Sub-características	Comportamiento en el tiempo
Objetivo	Lograr que el sistema ejecute correctamente las peticiones solicitadas por el usuario en menos de 5 segundo de tiempo.
Origen	El usuario
Artefacto	El usuario
Entorno	Funcionando correctamente
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a El usuario realiza una petición al sistema	
	El sistema responde correctamente la petición
Medida de respuesta	
3 segundos	

2.4.4. ESPECIFICACIÓN DEL REQUISITO NO FUNCIONAL: MANTENIBILIDAD

Tabla 13: Especificación del requisito no funcional Mantenibilidad -> Cambiabilidad

Atributo de Calidad	Mantenibilidad
Sub-atributos/Sub-características	Cambiabilidad
Objetivo	Facilitar la correcta modificación del sistema en caso de ser necesario.
Origen	El desarrollador
Artefacto	El sistema
Entorno	Funcionando correctamente / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a El desarrollador realiza una modificación en el sistema	
	El sistema asimila correctamente la modificación realizada.
Medida de respuesta	
NA	

Tabla 14: Especificación del requisito no funcional Mantenibilidad -> Cumplimiento de mantenibilidad

Atributo de Calidad	Mantenibilidad
Sub-atributos/Sub-características	Cumplimiento de mantenibilidad
Objetivo	Garantizar la actualización de las tecnologías de desarrollo.
Origen	El desarrollador
Artefacto	El sistema
Entorno	Funcionando correctamente / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a El desarrollador realiza una modificación en el sistema	

	El sistema asimila correctamente la modificación realizada.
Medida de respuesta	
NA	

2.4.5. ESPECIFICACIÓN DEL REQUISITO NO FUNCIONAL: PORTABILIDAD

Tabla 15: Especificación del requisito no funcional Portabilidad -> Adaptabilidad

Atributo de Calidad	Portabilidad
Sub-atributos/Sub-características	Adaptabilidad
Objetivo	Lograr que el sistema sea adaptable a diferentes entornos especificados.
Origen	El usuario
Artefacto	El sistema
Entorno	Funcionando correctamente
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a Se prueba el sistema en diferentes entornos.	
	El sistema funciona correctamente.
Medida de respuesta	
2 segundos	

Tabla 16: Especificación del requisito no funcional Portabilidad -> Instalabilidad

Atributo de Calidad	Portabilidad
Sub-atributos/Sub-características	Instalabilidad
Objetivo	Lograr que el sistema se pueda instalar en un entorno especificado.
Origen	El usuario
Artefacto	El sistema
Entorno	Funcionando correctamente / Modo degradado
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
1.a Se instala el sistema en diferentes entornos.	
	El sistema funciona correctamente.
Medida de respuesta	
NA	

2.4.6. ESPECIFICACIÓN DEL REQUISITO NO FUNCIONAL: RESTRICCIONES DEL DISEÑO

Tabla 17: Especificación del requisito no funcional Restricciones del diseño

Atributo de Calidad	No Aplica
Sub-atributos/Sub-características	Restricciones de diseño
Objetivo	Para el correcto funcionamiento del sistema se necesitan como tecnologías a utilizar: Python 2.7, Django 1.8, Bootstrap 4.0, Angular 5.2.10, PostgreSQL 9.4, PyCharm 2016.3.2, VSCode 1.18.1. Navegadores: Chrome 65.0, Firefox 59.2, IE 9,10,11, IE Mobile 11.
Origen	Externo al sistema

Artefacto	Código del sistema
Entorno	Operación normal
Estímulo	**Respuesta: Flujo de eventos (Escenarios)**
NA	NA

2.5. DIAGRAMA DE CLASES

2.5.1. DIAGRAMA DE PAQUETES

El diagrama del paquete muestra la disposición y organización de los elementos del modelo en proyectos de mediana a gran escala. Puede mostrar tanto la estructura como las dependencias entre subsistemas o módulos (*UML Modeling - Unified Modeling Language Tool* 2010). A continuación, en la figura 4 se muestra el diagrama de paquetes en el cual se agrupan los conceptos lógicos del problema y se muestra su relación.

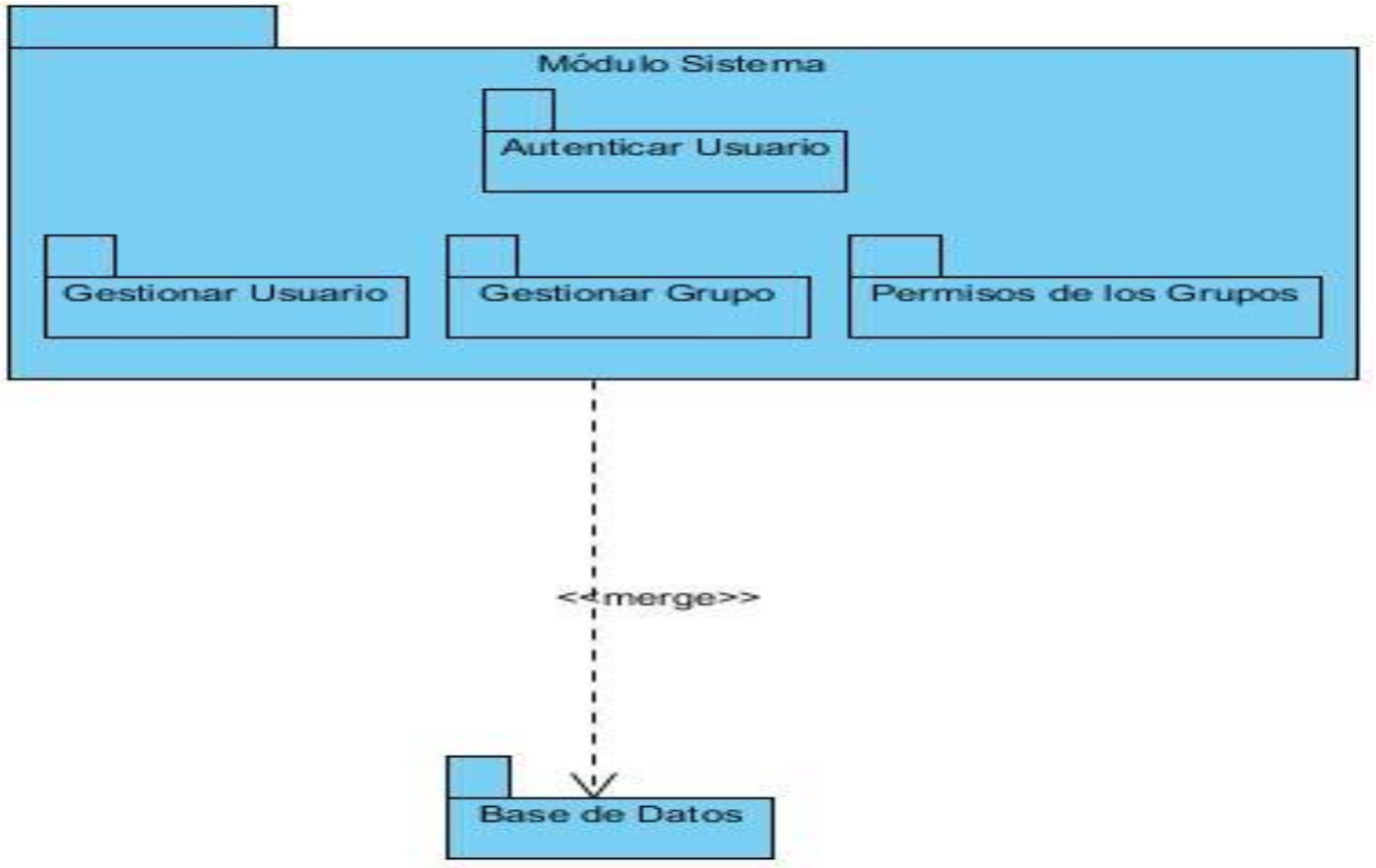


Figura 4: Diagrama de paquetes de Xilema base Web 2.0

2.5.2. DIAGRAMA DE CLASES

El diagrama de clases proporciona una visión general del sistema de destino al describir los objetos y las clases dentro del sistema y las relaciones entre ellos. Proporciona una amplia variedad de usos; desde modelar la estructura de datos específica del dominio hasta el diseño detallado del sistema objetivo (*UML Modeling - Unified Modeling Language Tool* 2010). A continuación, en la figura 5 se muestra el diagrama de clases referente a las entidades existentes en el sistema Xilema base Web 2.0.

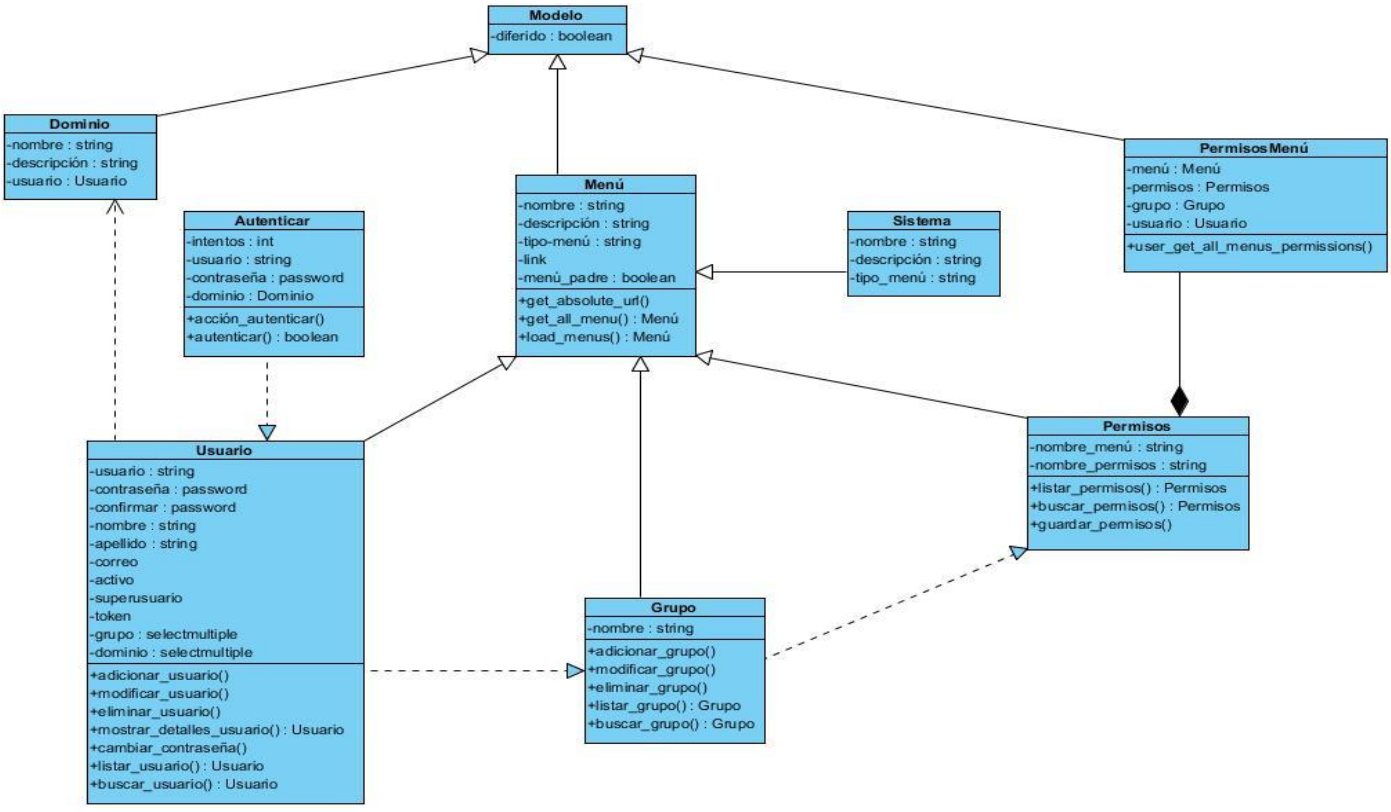


Figura 5: Diagrama de clases de Xilema base Web 2.0

2.5.3. DESCRIPCIÓN DE CLASES

Las descripciones de las clases se realizan con el fin de proporcionar información notable sobre cada una de las clases del sistema, tales como: número del módulo, número de la clase, nombre de la clase, propósito, una descripción visual de los atributos y métodos propios de la clase y observaciones que puedan ser relevantes. A continuación, se muestra la descripción de la clase Autenticar Usuario, el resto de las descripciones se encuentra en Anexo5.

2.5.3.1. CLASE AUTENTICAR USUARIO

Tabla 18: Descripción de la clase _ Autenticar usuario

Número del módulo	1
Número de la clase	1
Clase	Autenticar
Propósito	El usuario sea autenticado en el sistema.
Descripción	
Observaciones	

2.6. MODELO DE BASE DE DATOS

En el modelo de base de datos se determina la estructura lógica de una base de datos, el modo de almacenar, organizar y manipular los datos. Se utiliza para representar la base de datos de manera gráfica a través del diagrama de entidad relación (*Data Modeling - UML Diagramming Software* 2010). Los datos principales del sistema se encuentran en las tablas *auth_user, auth_group, auth_permission, app_menu* y *app_domain* las cuales tienen una relación de muchos a muchos por lo que se generan nuevas tablas manifestando esta relación. La tabla *auth_user* contiene los datos referentes a los usuarios del sistema dígase: usuario, contraseña, confirmar, nombre, apellido, correo, activo, superusuario, grupo y dominio. La tabla *auth_group* contiene los datos referentes a los grupos del sistema dígase: nombre. La tabla *auth_permission* contiene los datos referentes a los permisos del sistema dígase: nombre y tipo de permiso. A continuación, en la figura 6 se muestra el modelo de base de datos, donde se exponen los datos y la relación existente entre los mismo para el desarrollo del sistema Xilema base Web 2.0.

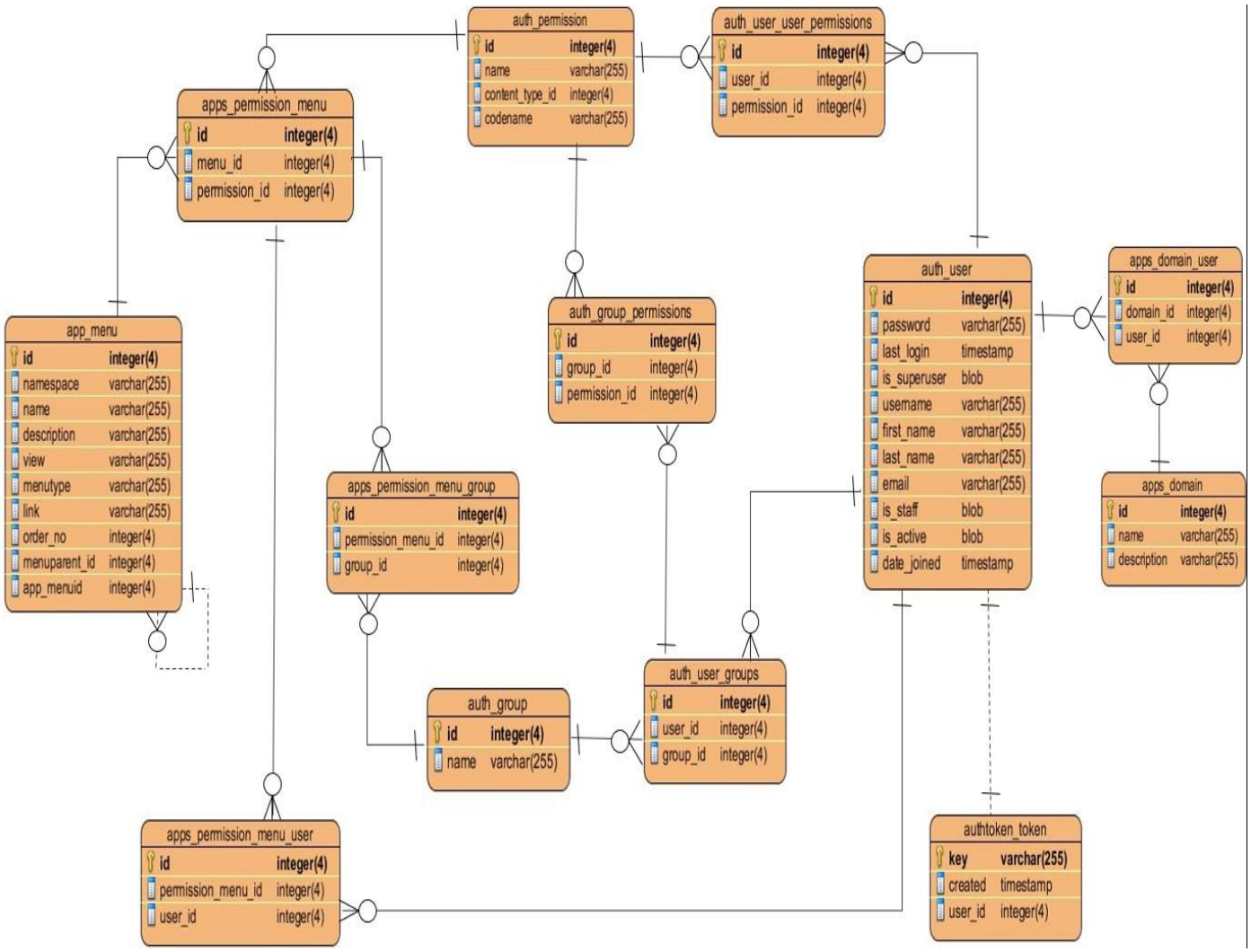


Figura 6: Modelo de base de datos de Xilema base Web 2.0

2.7. DIAGRAMA DE DESPLIEGUE

El diagrama de despliegue ayuda a modelar el aspecto físico de un sistema de software orientado a objetos. Modela la configuración del tiempo de ejecución en una vista estática y visualiza la distribución de componentes en una aplicación. En la mayoría de los casos, implica el modelado de las configuraciones de hardware junto con los componentes de software que perduraron (*UML Modeling - Unified Modeling Language Tool* 2010). En el diagrama de despliegue del sistema Xilema base Web 2.0 se puede apreciar como la estación de trabajo está compuesta por Web Browser. Los componentes *Presentation Layer* y *View Layer* pertenecientes al Angular Server envían toda la información generada al Web *Browser* a través de

http/https, asimismo el Angular Server se relaciona con el Python *Server* el cual se encarga de la relación de almacenamiento y obtención de datos con el *Database Server* a través del protocolo TCP/IP. A continuación, en la figura 7 se muestra el diagrama de despliegue en el cual se describe el despliegue físico de la información generada por el programa de software en los componentes de hardware.

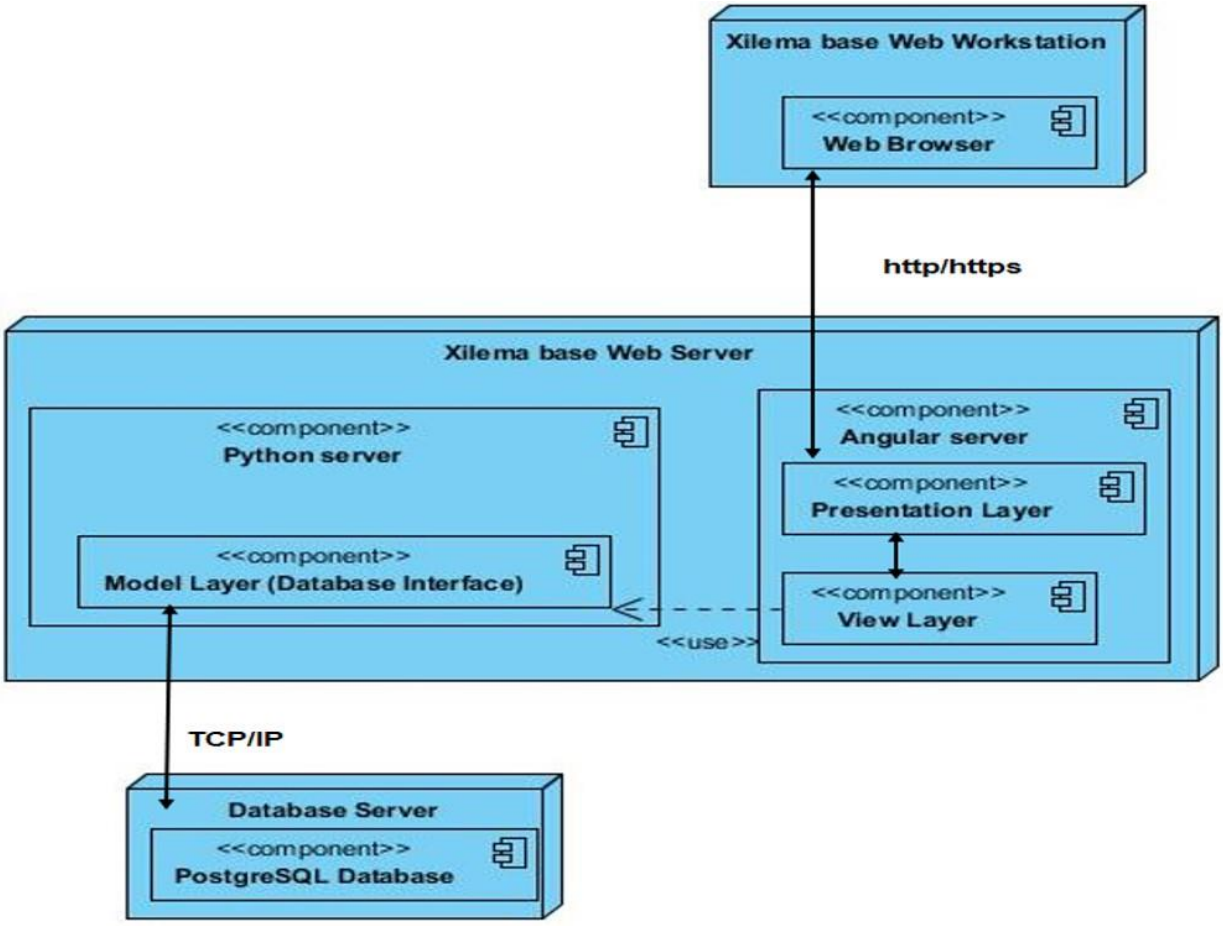


Figura 7: Diagrama de despliegue de Xilema base Web 2.0

2.8. CASOS DE PRUEBA

Los casos de prueba son un conjunto de condiciones o variables bajo las cuales se analiza que los requisitos funcionales de un sistema sean satisfactorios. Para desarrollar un caso de prueba se tienen en cuenta las siguientes características: escenario, descripción, todas las variables de entrada de datos al sistema, la respuesta del sistema y el flujo central. A continuación, se muestra las descripciones necesarias para

realizar el caso de prueba Autenticar Usuario, el resto de las descripciones referentes a los casos de prueba se encuentra en Anexo6.

2.8.1. DESCRIPCIÓN GENERAL: AUTENTICAR USUARIO.

El usuario llena los campos de texto mostrados por el sistema. El sistema valida los datos con los valores de la base de datos. Si los datos son correctos, el sistema autentica al usuario. Si los datos no son correctos, el sistema deniega al usuario mostrando una notificación donde se explica los motivos por el cual ha sido denegado. Luego el sistema permite al usuario volver a intentar autenticarse con los valores correctos

2.8.2. CONDICIONES DE EJECUCIÓN

El usuario se debe encontrar previamente almacenado en la base de datos.

2.8.3. SC AUTENTICAR USUARIO

Tabla 19: Caso de prueba _ Autenticar usuario

Escenario	Descripción	Variable usuario	Variable contraseña	Variable dominio	Respuesta del sistema	Flujo central
Autenticar	Autenticar los usuarios del sistema.	V wendy	V WTD	V local	El usuario ha sido autenticado.	Acceder a través de la dirección dónde se encuentra el Sistema. Llena el campo de texto usuario y contraseña. Seleccionar el dominio. Seleccionar el botón “Entrar”.
		I w3ndy	V WTD	V UCI.CU	Muestra notificación de valor inválido.	
		V wendy	I wendy	V local	Muestra notificación de valor inválido.	
		V wendy	I	V UCI.CU	Muestra notificación de campo vacío.	
		I	V WTD	V local	Muestra notificación de campo vacío.	
		V wendy	V WTD	I	Muestra notificación de seleccionar dominio.	

2.9. CONCLUSIONES PARCIALES

- ✓ Con la propuesta de solución quedaron expuestas las principales características necesarias para la realización del sistema.

- ✓ Con la obtención de los artefactos requeridos en la fase de ejecución de la metodología seleccionada se logró un correcto diseño del sistema.
- ✓ Con la realización de las descripciones de requisitos por procesos se logró un mejor entendimiento de las funcionalidades requeridas por el sistema. Asimismo, con la elaboración de las descripciones de los requisitos no funcionales.
- ✓ Con la elaboración de los casos de prueba se contribuyó a la realización de un correcto análisis sobre el funcionamiento del sistema.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DE VALIDACIÓN DEL PRODUCTO.

Con anterioridad se expusieron aspectos relacionados con las características que debe poseer la solución propuesta, de vital importancia para dar continuidad a la investigación. A continuación, se aborda la implementación del sistema referenciando los patrones arquitectónicos y de diseño definidos para el sistema, así como, los estándares de codificación a utilizar. En este capítulo se describen las pruebas que se pretenden realizar al sistema para validar su correcto funcionamiento, plasmando los resultados y los posibles artefactos que puedan generar las mismas.

3.1. PATRONES DE ARQUITECTURA DE SOFTWARE

La arquitectura de software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución (Reynoso 2004). Una arquitectura de software de un programa o un sistema computacional es la estructura del sistema, la cual comprende elementos de software, las propiedades externamente visibles de esos elementos, y las relaciones entre ellos (Bass, Clements, Kazman 2003). A continuación, se describen las características fundamentales de los patrones arquitectónicos MTV, cliente/servidor y basado en componentes, que serán utilizados para el desarrollo de la solución.

3.1.1. PATRÓN ARQUITECTÓNICO MODELO VISTA PLANTILLA

El objetivo de utilizar patrones como el MTV es principalmente hacer más fluida la comunicación entre desarrolladores, ya que permite organizar el trabajo de los mismos, favorece a crear independencia en el funcionamiento, permitiendo realizar cambios en una parte en particular sin afectar el resto de la aplicación. Además, facilita el mantenimiento en caso de errores y ofrece maneras más simples de demostrar el correcto desempeño de la aplicación. Separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.(Moss, Holovaty, Kaplan 2008)

El Modelo: se refiere a la capa de acceso de datos. Esta capa contiene todo lo respectivo a los datos, cómo acceder a ellos, cómo validarlos, qué comportamiento tienen y las relaciones entre ellos. En el sistema Xilema base Web el encargado de esta sección es el servidor de Python y Django, como se muestra en la figura 8.

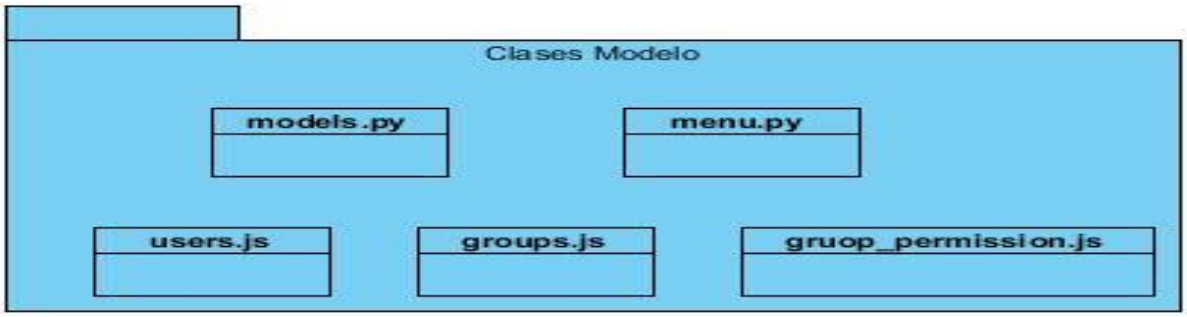


Figura 8: Capa modelo del patrón arquitectónico MTV

La Plantilla: se refiere a la capa de presentación. Esta capa contiene las decisiones relacionadas con la presentación de cómo debe mostrarse la información del sistema. En el sistema Xilema base Web el encargado de esta sección es el servidor de Angular, como se muestra en la figura 9.

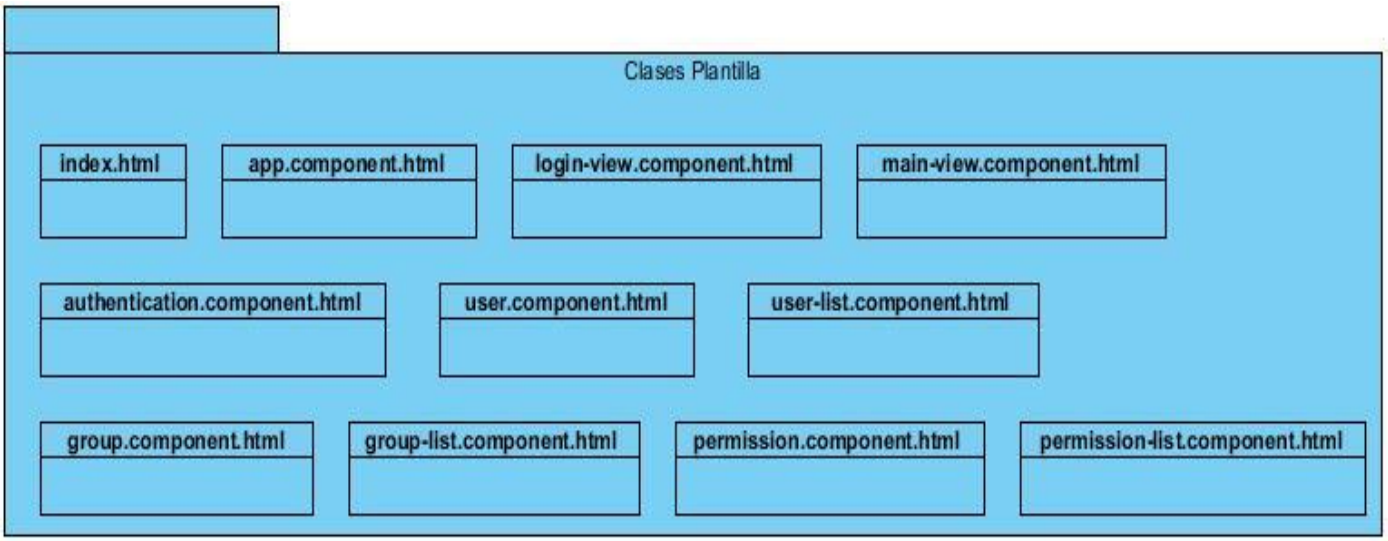


Figura 9: Capa plantilla del patrón arquitectónico MTV

La Vista: se refiere a la capa de la lógica de negocio. Esta capa contiene el acceso al modelo y delega en la plantilla apropiada esta información. En el sistema Xilema base Web el encargado de esta sección es el servidor de Angular, como se muestra en la figura 10.

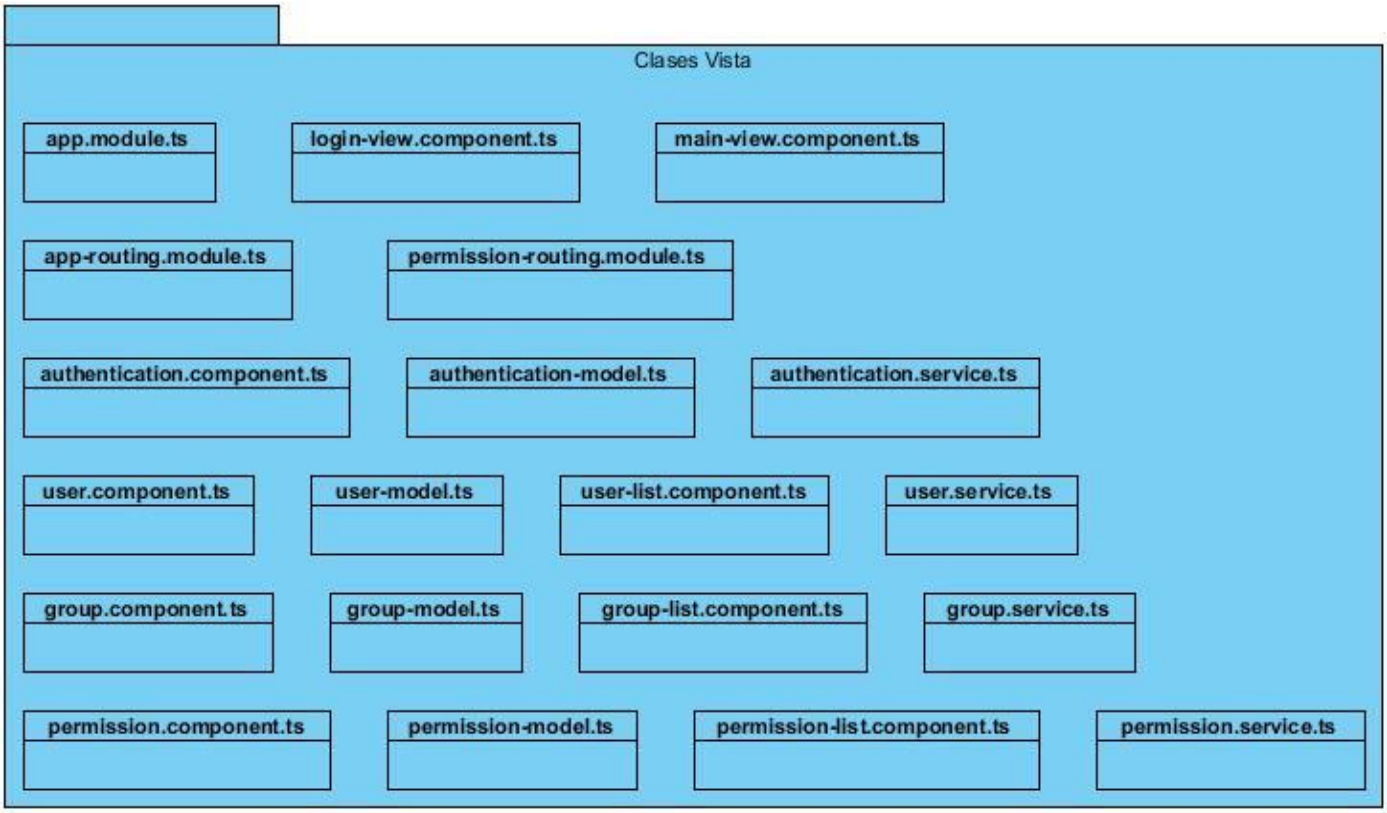


Figura 10: Capa vista del patrón arquitectónico MTV

3.1.2. PATRÓN ARQUITECTÓNICO CLIENTE SERVIDOR

La arquitectura Cliente/Servidor es el procesamiento cooperativo de la información por medio de un conjunto de procesadores, en el cual múltiples clientes, distribuidos geográficamente, solicitan requerimientos a uno o más servidores centrales. En este tipo de arquitectura el cliente puede ser desde uno hasta muchos consumidores del servicio que brinda un servidor distribuido en una misma computadora física o no.(Camacho, Cardeso, Nuñez 2004)

Para el desarrollo de Xilema base Web 2.0 se contará con un servidor desarrollado con las tecnologías de Python 2.7 y Django 1.8, el cual brindará los servicios necesarios que serán utilizados por el marco de trabajo de JavaScript empleado para el desarrollo de la interfaz de usuario Angular 5.2.10, como se muestra en la figura 11.

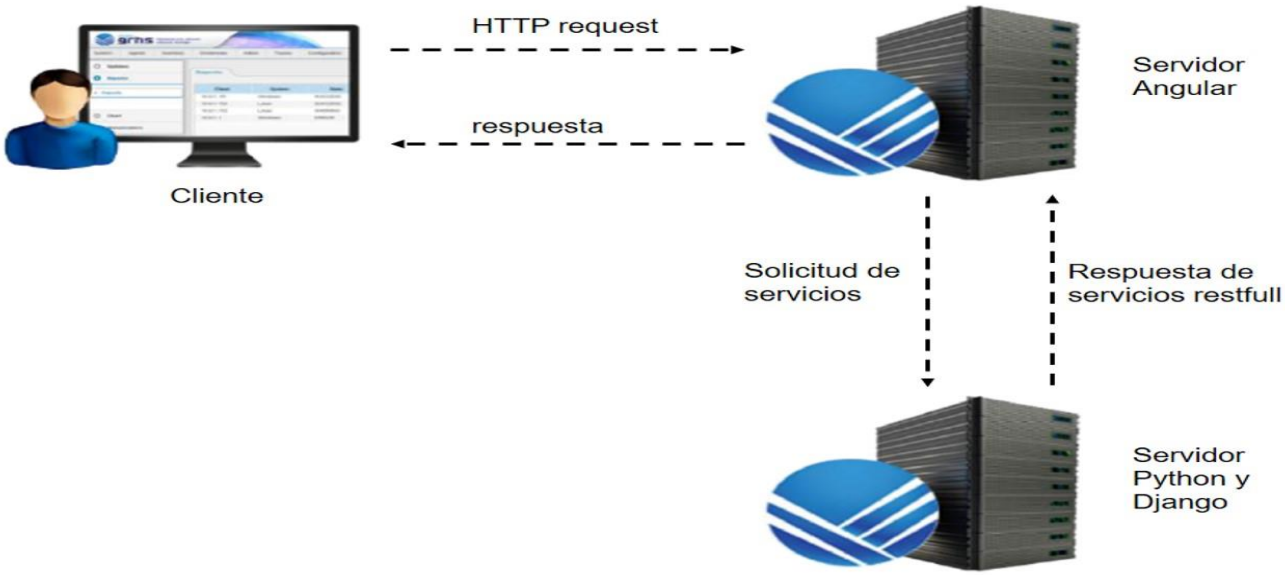


Figura 11: Patrón arquitectónico cliente-servidor

3.1.3. PATRÓN ARQUITECTÓNICO BASADO EN COMPONENTES

La arquitectura basada en componentes consiste en una rama de la ingeniería de software en la cual se trata con énfasis la descomposición del software en componentes funcionales. Esta descomposición permite convertir componentes pre-existentes en piezas más grandes de software. Este proceso de construcción de una pieza de software con componentes ya existentes, da origen al principio de reutilización del software, mediante el cual se promueve que los componentes sean implementados de una forma que permita su utilización funcional sobre diferentes sistemas en el futuro.(*Arquitectura Basada en Componentes* [no date])

Para el desarrollo de la interfaz de usuario se utiliza el marco de trabajo de JavaScript Angular el cual define un conjunto de componentes para el desarrollo del software, en el caso del sistema Xilema base Web se definieron como componentes *views* y *modules* que contienen un conjunto de subcomponentes, como se muestra en la figura 12.

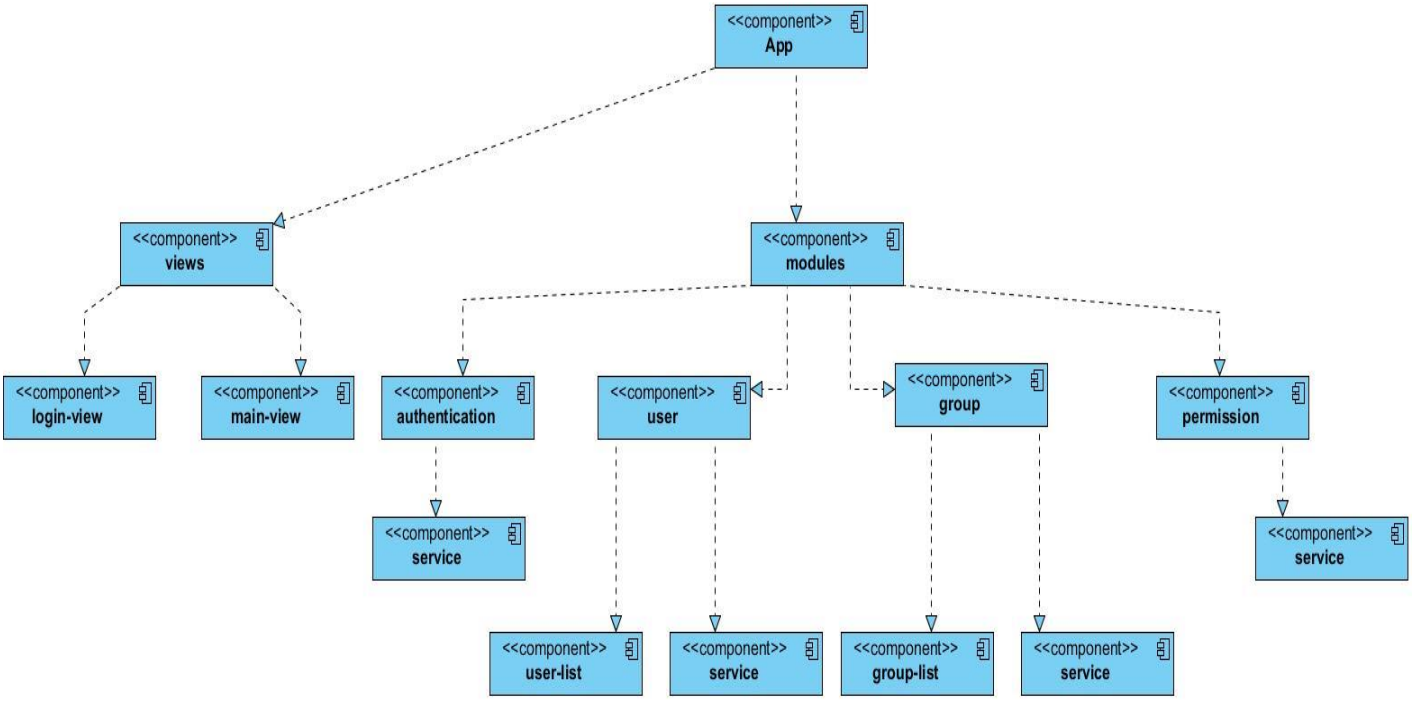


Figura 12: Patrón arquitectónico basado en componentes

3.2. PATRONES DE DISEÑO

En la ingeniería del software, un patrón constituye el apoyo para la solución a los problemas más comunes que se presentan durante las diferentes etapas del ciclo de vida del software. Los patrones de diseño describen soluciones simples y elegantes a problemas específicos en el diseño de software orientado a objetos, son combinaciones de componentes, casi siempre clases y objetos que por experiencia se sabe que resuelven ciertos problemas de diseño comunes (Guerrero, Suárez, Gutiérrez 2013). Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software (Tedeschi 2018). A continuación, se describen los patrones de diseño que serán utilizados para el desarrollo de la solución.

3.2.1. PATRONES GENERALES DE SOFTWARE PARA ASIGNAR RESPONSABILIDADES

GRASP es un acrónimo de *General Responsibility Assignment Software Patterns* (patrones generales de software para asignar responsabilidades). Los patrones GRASP constituyen un apoyo para la enseñanza que ayuda a en tender el diseño de objetos esencial, y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. Este enfoque para la comprensión y utilización de los principios de diseño

se basa en los patrones de asignación de responsabilidades (Larman [no date]). A continuación, se describen los patrones generales de software para asignar responsabilidades, que serán utilizados para el correcto desarrollo de la solución.

3.2.1.1. EXPERTO

Solución: asignar una responsabilidad al experto en información: la clase que posee la información necesaria para cumplir con la responsabilidad.(Larman [no date])

El patrón experto se evidencia en las clases correspondientes al servidor de Python y Django, dichas clases son: *Menu, Domain, Permission_Menu.*

3.2.1.2. CREADOR

Solución: asignar a la clase B la responsabilidad de crear una instancia de clase A si se cumple uno o más de los casos siguientes: B agrega objetos de A, B contiene objetos de A, B registra instancias de objetos de A, B utiliza más estrechamente objetos de A, B tiene los datos de inicialización que se pasarán a un objeto de A cuando sea creado.(Larman [no date])

El patrón creador se evidencia en las clases referentes a los componentes *authentication, user* y *group* en el servidor correspondiente a Angular, dichas clases son: *AuthenticationComponent, GroupComponent, UserComponent,*

3.2.1.3. CONTROLADOR

Solución: asignar la responsabilidad de recibir o manejar un mensaje de evento del sistema a una clase que representa una de las siguientes opciones: representa el sistema global, dispositivo o subsistema y representa un escenario de caso de uso en el que tiene lugar el evento del sistema.(Larman [no date])

El patrón controlador se evidencia en las clases referentes al subcomponente *service* de los componentes *authentication, user* y *group* en el servidor correspondiente a Angular, dichas clases son: *AuthenticationService, UserService, GroupService.*

3.2.1.4. BAJO ACOPLAMIENTO

Solución: asignar una responsabilidad de manera que el acoplamiento permanezca bajo. El acoplamiento es una medida de la fuerza con que un elemento está conectado a, tiene conocimiento de, confía en, otros

elementos. Un elemento con bajo (o débil) acoplamiento no depende de demasiados otros elementos. Estos elementos pueden ser clases, subsistemas, sistemas, entre otros.(Larman [no date])

El patrón bajo acoplamiento se evidencia en las clases referentes al componente app y los subcomponentes *login-view* y *main-view* en el servidor correspondiente a Angular, dichas clases son: *AppComponent, LoguinViewComponent, MainViewComponent.*

3.2.1.5. ALTA COHESIÓN

Solución: asignar una responsabilidad de manera que la cohesión permanezca alta. En cuanto al diseño de objetos, la cohesión es una medida de la fuerza con la que se relacionan y del grado de focalización de las responsabilidades de un elemento. Un elemento con responsabilidades altamente relacionadas, y que no hace una gran cantidad de trabajo, tiene alta cohesión. Estos elementos pueden ser clases, subsistemas, entre otros.(Larman [no date])

El patrón alta cohesión se evidencia a la hora de organizar las responsabilidades de cada clase, implementando las funcionalidades necesarias para cada una de ellas.

3.2.2. GANG OF FOUR

Los patrones de diseño según *The Gang of Four* (GOF) describen soluciones simples y elegantes a problemas específicos en el diseño de software orientado a objetos. GOF presenta un conjunto de 23 patrones de diseño los cuales se clasifican y agrupan a partir de dos criterios, su propósito y alcance, las categorías definidas son: creacionales, estructurales y de comportamiento (Guerrero, Suárez, Gutiérrez 2013). A continuación, se describen los patrones GOF, que serán utilizados para el correcto desarrollo de la solución.

3.2.2.1. CREACIONALES

Los patrones creacionales se ocupan del proceso de creación de clases y objetos, son los encargados de abstraer el proceso de instanciación o creación de objetos, ayudan a que el sistema sea independiente de cómo sus objetos son creados, integrados y representados (Guerrero, Suárez, Gutiérrez 2013). Los patrones que hacen parte de esta categoría son: método de fabricación, fabricación abstracta, constructor, prototipo e interfaz única. A continuación, se describen los patrones creacionales empleados en el desarrollo del sistema Xilema base Web:

- ***Constructor:*** abstrae el proceso de creación de un objeto complejo, centralizando dicho proceso en un único punto. La utilización de este patrón se ve presente en los constructores referentes a las clases de los componentes del sistema.

3.2.2.2. ESTRUCTURALES

Los patrones estructurales tratan de la composición de clases y objetos, se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes Los patrones en esta categoría se encargan de lograr que los cambios en los requisitos de la aplicación no ocasionen cambios en las relaciones entre los objetos (Guerrero, Suárez, Gutiérrez 2013). Los patrones que hacen parte de esta categoría son: adaptador, puente, compuesto, decorador, fachada, *flyweight* y proxy. A continuación, se describen los patrones estructurales empleados en el desarrollo del sistema Xilema base Web:

- ***Decorador:*** permite añadir funcionalidades y responsabilidades a un objeto de forma dinámica, facilitando una alternativa muy flexible a la creación de subclases para extender funcionalidades. La utilización de este patrón se ve presente al utilizar *@Component* y *@Input* en los componentes usuario, grupo y permisos de grupos y *@Injectable* en los servicios de cada componente.

3.2.2.3. COMPORTAMIENTO

Los patrones de comportamiento caracterizan las formas en que las clases o los objetos interactúan y distribuyen la responsabilidad. Son los encargados de las opciones de comportamiento de la aplicación, permitiendo que el comportamiento varíe en tiempo de ejecución, sin estos patrones cada comportamiento tendría que diseñarse e implementarse por separado (Guerrero, Suárez, Gutiérrez 2013). Los patrones que se agrupan en esta categoría son: interprete, método de plantilla, cadena de responsabilidad, comando, iterador, mediador, recuerdo, observador, estado, estrategia y visitante. A continuación, se describen los patrones de comportamiento empleados en el desarrollo del sistema Xilema base Web:

- ***Método de plantilla:*** define en una operación el esqueleto del algoritmo, delegando en las subclases algunos de sus pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar su estructura. La utilización de este patrón se ve presente en los componentes *views: login-view* y *main-view*, del sistema.
- ***Iterador:*** define una interfaz que declara los métodos necesarios para acceder secuencialmente a un grupo de objetos de una colección. Permite realizar recorridos sobre objetos compuestos

independiente de la implementación de estos. La utilización de este patrón se ve presente en la directiva *ngFor empleadas para mostrar los datos en las tablas de listados.

- ***Observador:*** define una relación de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. La utilización de este patrón se ve presente en una de las facilidades que brinda la tecnología seleccionada Angular que es el enlazado de datos en tiempo real.

3.3. ESTÁNDARES DE CODIFICACIÓN

Se definen estándares de codificación porque un estilo de programación homogéneo en un proyecto permite que todos los participantes lo puedan entender en menos tiempo y que el código en consecuencia sea mantenibles (Manuel Arias Calleja 2008). A continuación, se describen los estándares de codificación para los archivos, selectores, clases, atributos, métodos y variables, definidos por el marco de trabajo Angular para el desarrollo de sus aplicaciones.

3.3.1. NOMENCLATURA DE ARCHIVOS

Los nombres de los archivos se escribirán con minúscula y se separarán con guiones **"-"** si en la parte descriptiva del nombre y con puntos **"."** en la parte descriptiva del tipo, por ejemplo:

main-page.component.html ***main-page.component.ts*** ***app-routing.module.ts***

app.component.html ***app.component.ts*** ***app.module.ts***

3.3.2. NOMENCLATURA DE SELECTORES DE COMPONENTES

Los nombres de los selectores de los componentes se escribirán entre comillas simples **' '** , se iniciarán siempre con el nombre **"app"** y se separarán con guiones **"-"** *el resto del nombre descriptivo del selector, por ejemplo:*

'app-root' ***'app-auth-page'*** ***'app-main-page'*** ***'app-forms'*** ***'app-forms-user'***

3.3.3. NOMENCLATURA DE CLASES

Los nombres de las clases se escribirán con letra inicial mayúscula y si es compuesto en la parte descriptiva del nombre iniciará con mayúscula cada una de las palabras subsiguientes, luego se escribirá con letra inicial mayúscula el nombre del símbolo referente al nombre del archivo, por ejemplo:

class MainPageComponent ***class AppComponent*** ***class AppRoutingModule***

class AuthPageComponent ***class FormsComponent***

3.3.4. NOMENCLATURA DE ATRIBUTOS

Los nombres de los atributos se escribirán con letra inicial minúscula y si es compuesto iniciará con mayúscula cada una de las palabras subsiguientes, por ejemplo:

userName ***userEmail*** ***user*** ***groupName*** ***group*** ***permissions***

3.3.5. NOMENCLATURA DE MÉTODOS

Los nombres de los métodos se escribirán con letra inicial minúscula y si es compuesto iniciará con mayúscula cada una de las palabras subsiguientes, luego seguidos de paréntesis **"( )"** con sus respectivos parámetros y las llaves **"{ }"** para el cuerpo del método, por ejemplo:

getName(){…} ***getDomain(){…}*** ***setEmail(){…}***

3.3.6. NOMENCLATURA DE VARIABLES

Los nombres de las variables se escribirán completamente en minúscula o mayúscula y si el nombre es compuesto se separarán las distintas palabras con guiones bajos **"_"**, por ejemplo:

domain ***group*** ***GROUP_PERMISSIONS***

3.4. PRUEBAS DE VALIDACIÓN

Las pruebas de validación son un proceso de revisión al que se somete un programa informático para comprobar que cumple con sus especificaciones. El mismo, que suele tener lugar al final de la etapa de

desarrollo, se realiza principalmente con la intención de confirmar que la aplicación permita llevar a cabo las tareas que sus potenciales usuarios esperan.(Porto, Gardey 2013)

Las pruebas de validación comienzan en la culminación de las pruebas de integración, cuando se ejecutaron componentes individuales, el software está completamente ensamblado como un paquete y los errores de interfaz se descubrieron y corrigieron. Las pruebas se enfocan en las acciones visibles para el usuario y las salidas del sistema reconocibles por el usuario. La validación puede definirse en muchas formas, pero una definición simple es que la validación es exitosa cuando el software funciona en una forma que cumpla con las expectativas razonables del cliente (Pressman 2010). A continuación, se describen las pruebas de validación que se realizarán al sistema Xilema base Web 2.0. Se tendrán dos estrategias de prueba, las pruebas de caja blanca, las cuales van dirigidas al código y las pruebas de caja negra, las cuales van dirigidas al cliente.

3.4.1. PRUEBAS DE CAJA BLANCA

Las pruebas de caja blanca del software se basan en el examen cercano de los detalles de procedimiento. Las rutas lógicas a través del software y las colaboraciones entre componentes se ponen a prueba al revisar conjuntos específicos de condiciones y/o bucles (Pressman 2010). Las pruebas de caja blanca son pruebas dirigidas al código, las cuales se realizan de forma automática por las herramientas especializadas en este tipo de prueba. El tipo de prueba de caja blanca a ejecutar son las pruebas unitarias.

3.4.1.1. PRUEBAS UNITARIAS

Las pruebas unitarias enfocan los esfuerzos de verificación en la unidad más pequeña del diseño de software: el componente o módulo de software. La relativa complejidad de las pruebas y los errores que descubren están limitados por el ámbito restringido que se establece para la prueba. Las pruebas unitarias se enfocan en la lógica de procesamiento interno y de las estructuras de datos dentro de las fronteras de un componente. Este tipo de pruebas puede realizarse en paralelo para múltiples componentes (Pressman 2010). Una prueba unitaria es un método que puede invocar al código que se desea probar, donde se determina si el resultado obtenido es el esperado. Si es igual, entonces la prueba es exitosa.

A continuación, en las figuras 13,14 y 15 se muestran los resultados obtenidos durante la aplicación de dichas pruebas a la aplicación que se desarrolla. Se realizan 3 iteraciones para 15 fragmentos de código, sobre los que se detectan 12 errores que se resuelven de manera satisfactoria. Las pruebas se realizan

utilizando como herramienta Jazmín y Karma y pueden consultarse escribiendo en la consola ***npm t*** o en los archivos ****.spec.ts*** referentes a cada componente.

Iteración 1: en esta iteración se seleccionan 5 fragmentos de código en los cuales se detectan 4 errores de lógica algorítmica, que se resuelven de manera satisfactoria.

Figura 13: Prueba unitaria iteración 1

Iteración 2: en esta iteración se seleccionan 5 fragmentos de código en los cuales se detectan 3 errores de lógica algorítmica, que se resuelven de manera satisfactoria.

Figura 14:Prueba unitaria iteración 2

Iteración 3: en esta iteración se seleccionan 5 fragmentos de código en los cuales se detectan 5 errores de lógica algorítmica, que se resuelven de manera satisfactoria.

Figura 15:Prueba unitaria iteración 3

3.4.2. PRUEBAS DE CAJA NEGRA

Las pruebas de caja negra se refieren a las pruebas que se llevan a cabo en la interfaz del software. Una prueba de caja negra examina algunos aspectos fundamentales de un sistema con poca preocupación por la estructura lógica interna del software (Pressman 2010). Las pruebas de caja negras son pruebas dirigidas al cliente, en las cuales se mide el nivel de satisfacción del cliente. El tipo de prueba de caja negra a ejecutar son las pruebas de aceptación.

3.4.2.1. PRUEBAS DE ACEPTACIÓN

Las pruebas de aceptación, también llamadas pruebas funcionales son supervisadas por el cliente basándose en los requerimientos tomados de las descripciones de requisitos funcionales. Son pruebas de caja negra, que representan un resultado esperado de determinada transacción con el sistema. Para que una descripción de requisito funcional se considere aprobada, deberá pasar todas las pruebas de aceptación elaboradas para dicha descripción (Joskowick 2008). Las pruebas de aceptación se realizan para evaluar el grado de calidad del software de acuerdo a todos los aspectos relevantes que intervienen en el sistema.

Las pruebas de aceptación se realizan a partir de los casos de pruebas anteriormente desarrollados para cada escenario de las descripciones de los requisitos funcionales. Al llevar a cabo las pruebas para el sistema, se realizaron 3 iteraciones para 9 pruebas de aceptación, donde se obtuvieron 3 no conformidades, las cuales fueron resueltas en su totalidad. En la primera iteración se realizaron 4 pruebas de aceptación y fueron detectadas 2 no conformidades. En la segunda iteración se realizaron 3 pruebas de aceptación y fue detectada 1 no conformidad. En la tercera iteración se realizaron 2 pruebas de aceptación y no fue detectada ninguna no conformidad. A continuación, en la figura 16 se muestran los resultados de las pruebas obtenidos para cada iteración.

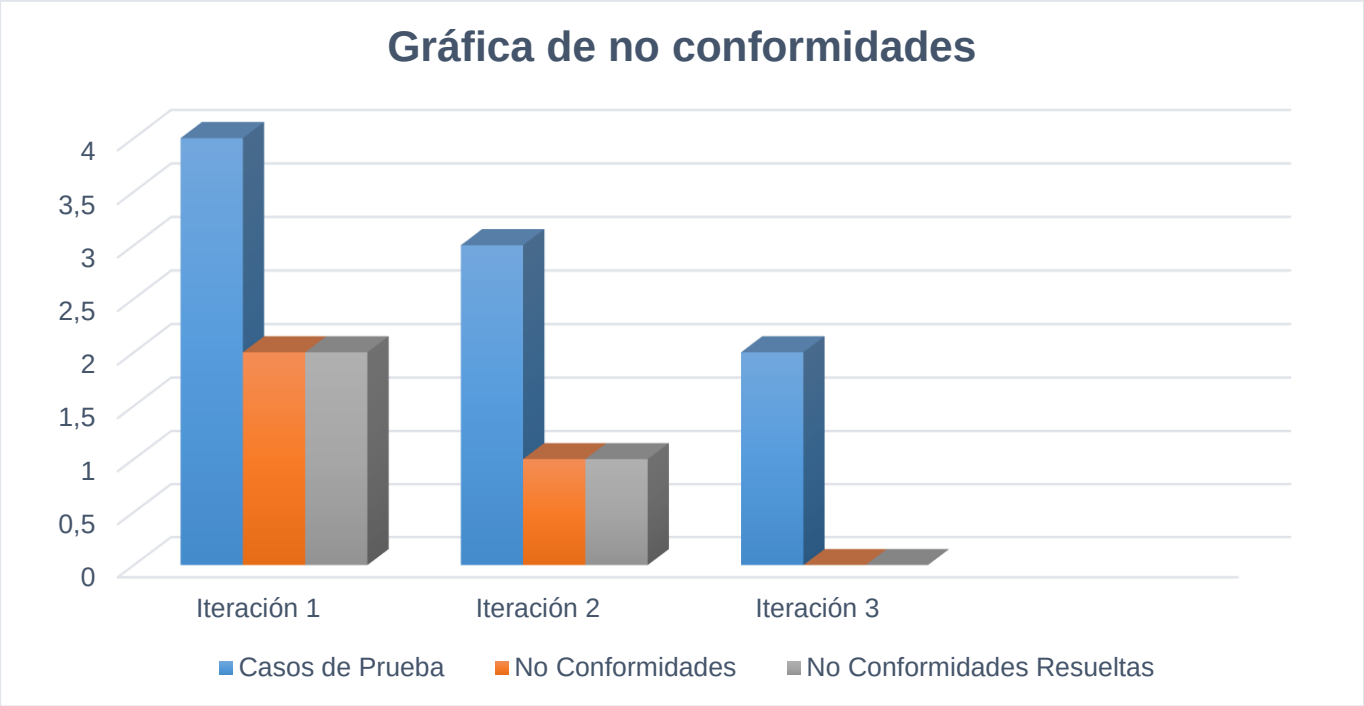


Figura 16: Gráfica de comportamiento de no conformidades

A continuación, en la tabla 20 se muestran las descripciones de cada una de las no conformidades que fueron detectadas por el cliente.

Tabla 20: Descripción de no conformidades encontradas.

Nº	Descripción	Iteración
1.	El submenú Gestión de Usuario esta fijo, no se despliega para mostrar u ocultar las opciones Usuarios, Grupos y Permisos de grupos.	1
2.	En la opción Usuarios falta el botón referente a cambiar contraseña de un usuario.	1
3.	Los formularios salen debajo de la tabla de listados y deben salir en ventanas emergentes.	2

3.5. CONCLUSIONES PARCIALES

- ✓ Con la definición de los patrones arquitectónicos se estableció la estructura del sistema y los principios que orientan su diseño y evolución.

- ✓ Con la utilización de los patrones de diseño se contribuyó a facilitar el adecuado desarrollo del software.
- ✓ La utilización de los estándares de codificación permitió la elaboración del código del sistema de manera limpia y organizada, siendo posible el entendimiento para cualquier programador que utilice el código desarrollado.
- ✓ Con las pruebas de validación del software se corroboró el correcto funcionamiento de las descripciones de requisitos funcionales implementadas.

CONCLUSIONES

Se describió con gran nivel de detalles la migración del sistema Xilema base Web en su versión 2.0 a partir de las nuevas tecnologías de desarrollo seleccionadas. Una vez finalizada la investigación se concluye que:

- ✓ El estudio de los marcos de trabajo que utilizan JavaScript con mayor demanda en el mercado laboral evidenció las características, así como las ventajas y las desventajas de cada uno de los marcos de trabajo, y permitió definir la factibilidad de la propuesta de solución.
- ✓ La selección de las herramientas utilizadas en el desarrollo de la migración del sistema se orientó a las necesidades del cliente y restricciones de diseño establecidas.
- ✓ Se cumplió con las exigencias dictadas por la metodología de desarrollo del software AUP-UCI acerca de las fases y prácticas a efectuar para lograr un producto de calidad, generando los artefactos requeridos para el desarrollo, diseño e implementación del sistema Xilema base Web 2.0.
- ✓ Se desarrolló la propuesta de solución basada en las necesidades del cliente y el estudio realizado sobre los marcos de trabajo actuales que utilizan JavaScript.
- ✓ El uso de patrones arquitectónicos y de diseño brindó como resultado un diseño sólido y flexible para la codificación de la aplicación.
- ✓ Las pruebas realizadas al sistema corroboraron el buen funcionamiento del mismo de acuerdo con las descripciones de requisitos por procesos implementadas.

De manera general, se migró a la versión 2.0 del sistema Xilema base Web, permitiendo mejorar la usabilidad, eficiencia y manteniendo las restricciones de diseño establecidas por el cliente, concluyendo que se ha cumplido satisfactoriamente con los objetivos propuestos.

RECOMENDACIONES

Teniendo en cuenta los resultados obtenidos en la implementación de la solución se recomienda:

- Realizar la internacionalización del sistema Xilema base Web versión 2.0.
- Utilizar como base para la posterior migración de XilemaGRHS la propuesta para el *Backend* desarrollada por el presente trabajo de diploma.
- Implementar los restantes módulos del sistema XilemaGRHS sobre la nueva versión de Xilema base Web, basándose en el uso de las nuevas tecnologías.
- Realizar una aplicación Android que consuma los servicios brindados por lo nuevos servidores del sistema Xilema base Web versión 2.0.

REFERENCIA

DÍAZ LAZO, Juliet, PÉREZ GUTIÉRREZ, Adriana and FLORIDO BACALLAO, René. IMPACTO DE LAS TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES (TIC) PARA DISMINUIR LA BRECHA DIGITAL EN LA SOCIEDAD ACTUAL. *Cultivos Tropicales*. March 2011. Vol. 32, no. 1, p. 81–90.

AZAÑA, Daniel López. ¿Qué es la arquitectura web? [online]. 17 September 2014. [Accessed 4 December 2017]. Available from: http://www.daniloaz.com/es/que-es-la-arquitectura-web/

INSTITUTO TECNOLÓGICO DE MATEHUALA. 2.1 Arquitectura de las aplicaciones Web | Programación Web. [online]. 2015. [Accessed 4 December 2017]. Available from: https://programacionwebisc.wordpress.com/2-1-arquitectura-de-las-aplicaciones-web/

IBÁÑEZ, Carlos García, BALLESTEROS, Raqueñ Hervás and GERVÁS, Pablo. Una arquitectura software para el desarrollo de aplicaciones de generación de lenguaje natural. [online]. 2004. Vol. 33. Available from: http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/3065

Productos | Universidad de las Ciencias Informáticas. [online]. 2018. [Accessed 6 February 2018]. Available from: http://www.uci.cu/investigacion-y-desarrollo/productos

Centro de Telemática (TLM) | Universidad de las Ciencias Informáticas. [online]. 2018. [Accessed 6 February 2018]. Available from: http://www.uci.cu/investigacion-y-desarrollo/centros-de-desarrollo/centro-de-telematica-tlm

SHACKEL, Brian and RICHARDSON, Simon. *Human Factors for Informatics Usability* [online]. New York : Cambridge University Press, 1991. ISBN 0 521 36570 8. Available from: https://books.google.com.cu/books?hl=es&lr=&id=KSHrPgLlMJIC&oi=fnd&pg=PA21&dq=definition+of+usability&ots=IWPtOZ-SB9&sig=-TRQ_17PCizZLGV5qIPfMX5E-Xk&redir_esc=y#v=onepage&q&f=false

NERI, Carlos. No todo es click. Usabilidad, accesibilidad y experiencia del usuario en la Web. *Scribd* [online]. 2007. [Accessed 9 April 2018]. Available from: https://es.scribd.com/document/51645567/el-concepto-de-usabilidad

RAE- ASALE. eficiencia. *Diccionario de la lengua española - Edición del Tricentenario* [online]. 2018. [Accessed 29 May 2018]. Available from: http://dle.rae.es/?id=EPVwpUD

INTERNATION DESIGN FOUNDATION. Efficiency. *The Glossary of human Computer Interaction* [online]. 2002 2018. [Accessed 29 May 2018]. Available from: https://www.interaction-design.org/literature/book/the-glossary-of-human-computer-interaction/efficiency

RAE- ASALE. restricción. *Diccionario de la lengua española - Edición del Tricentenario* [online]. 2018. [Accessed 29 May 2018]. Available from: http://dle.rae.es/?id=WEgzpII

OXFORD DICTIONARIES. restriction. [online]. 2018. [Accessed 29 May 2018]. Available from: https://en.oxforddictionaries.com/definition/restriction

RAE- ASALE. diseño. *Diccionario de la lengua española - Edición del Tricentenario* [online]. 2018. [Accessed 29 May 2018]. Available from: http://dle.rae.es/?id=DuKP0H9

OXFORD DICTIONARIES DEFINITION. desig. [online]. 2018. [Accessed 29 May 2018]. Available from: https://en.oxforddictionaries.com/definition/design

VILÁ, Fermí. *JavaScript Manual* [online]. 2001. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/Javascript.pdf

PÉREZ, Javier Eguíluz. *Introducción a JavaScript* [online]. 2009. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/introduccion_javascript.pdf

SANDERS, william. *Learning PHP design patterns* [online]. O'Reilly Media, Inc, 2013. ISBN 978-1-4493-4491-7. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/learning_php_design_patterns.pdf

PÉREZ, Javier Eguíluz. *CSS avanzado* [online]. 2009. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/css_avanzado.pdf

PÉREZ, Javier Eguíluz. *Introducción a AJAX* [online]. 2008. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/introduccion_ajax.pdf

GARRETT, Jesse James. Ajax: A New Approach to Web Applications | Adaptive Path. [online]. 18 February 2015. [Accessed 21 March 2018]. Available from: http://adaptivepath.org/ideas/ajax-new-approach-web-applications/

NILESH, Jain, PRIYANKA, Mangal and DEEPAK, Mehta. AngularJS: A Modern MVC Framework in JavaScript. [online]. 2014. Vol. 5, no. 12. Available from: https://pdfs.semanticscholar.org/a8bd/ab90c7b568755d718f9fd0a92221488f757d.pdf

ANGULAR. Angular (@angular) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/angular

GACKENHEIMER, Cory. *Introduction to React. Using creact to build scalable and efficient user interfaces.* [online]. Editorial Board, 2015. ISBN 978-1-4842-1245-5. Available from: https://books.google.com.cu/books?id=NZCKCgAAQBAJGoogle-Books-ID: NZCKCgAAQBAJ

AM, Vipul and SONPATKI, Prathamesh. *ReaactJS by Example - Building Modern Web Applications with React* [online]. Packt Publishing Birmingham, 2016. ISBN 978-1-78528-964-4. Available from: http://droppdf.com/

REACT. React (@reactjs) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/reactjs

Introduction — Vue.js. *oficial* [online]. 2018. [Accessed 21 January 2018]. Available from: https://vuejs.org/v2/guide/Vue.js - The Progressive JavaScript Framework

KURNIAWAN, Agus. *Vue.js Programming by Example* [online]. 2017. Available from: https://books.google.es/books?id=JDw8DwAAQBAJGoogle-Books-ID: JDw8DwAAQBAJ

VUE.JS. Vue.js (@vuejs) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/vuejs

PURI, Suchit. *Ember.js Web Development with Ember CLI* [online]. Packt Publishing Birmingham, 2015. ISBN 978-1-78439-584-1. Available from: http://droppdf.com/

CRAVENS, Jesse and BRADY, Thomas Q. *Building Web Apps with Ember.js. Write ambitious JavaScript* [online]. O'Reilly Media, Inc., 2014. ISBN 978-1-4493-7092-3. Available from: http://oreilly.com/catalog/errata.csp?isbn=9781449370923

EMBERJS. EmberJS (@emberjs) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/emberjs

GUILBAULT, Manuel. *Learning Aurelia. Harness the power of the next-generation JavaScript Framework, Aurelia, and start creating apps that really set you apart.* [online]. Packt Publishing Birmingham, 2016. ISBN 978-1-78588-967-7. Available from: http://droppdf.com/

AURELIA. Aurelia (@AureliaEffect) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/aureliaeffect

GOOGLE. Google Trends. *Google Trends* [online]. 2018. [Accessed 30 January 2018]. Available from: https://trends.google.es/trends/explore?date=today%205-y&geo=ES&q=angular,react,vue.js,ember.js,aureliaDemanda de los frameworks Angular, React, VueJS, EmberJS, Aurelia

DARWIN, Peter Bacon and KOZLOWSKI, Pawel. *Mastering Web Application Development with AngularJS*. Packt Publishing, 2013.

BHAUMIK, Snig. *Bootstrap Essentials. Use the powerful features of bootstrap to create responsive and appealing web pages.* [online]. Packt Publishing Birmingham, 2015. ISBN 978-1-78439-517-9. Available from: http://droppdf.com/

What is Python? Executive Summary. *Python.org* [online]. 2001 2018. [Accessed 12 December 2017]. Available from: https://www.python.org/doc/essays/blurb/

HOLOVATY, Adrian, MOSSS, Jacob Kaplan and M, Saul Garcia. *La guía definitiva de Django. Desarrolla aplicaciones Web de forma rápida y sencilla.* [online]. Django Software Corporation, 2015. ISBN MA 02111-1307. Available from: http://github.com/saulgm/djangobook.com

ALFONSO, J Minguillón. *Introducción al Lenguaje de Modelado Unificado* [online]. FUOC. Available from: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/9121/1/Intro_UML.pdf

CORNEJO, Jose Enrrique González. El Lenguaje de Modelado Unificado (UML). [online]. January 2008. [Accessed 15 December 2017]. Available from: http://www.docirs.com/uml.htm

Visual Paradigm 8.0 (formerly VP-UML 8.0) Released. *Visual Paradigm* [online]. 16 August 2010. [Accessed 15 December 2017]. Available from: https://www.visual-paradigm.com/aboutus/newsreleases/vpuml80.jsp

What is PostgreSQL. *PostgreSQL tutorial* [online]. 2018. [Accessed 15 December 2017]. Available from: http://www.postgresqltutorial.com/what-is-postgresql/

Features - PyCharm. *Python IDE for Professional Developers* [online]. 2000 2018. [Accessed 15 December 2017]. Available from: https://www.jetbrains.com/pycharm/features/

Documentation for Visual Studio Code. *Visual Studio Code* [online]. 2018. [Accessed 21 March 2018]. Available from: https://code.visualstudio.com/doc

Git version control. *Git* [online]. 2018. [Accessed 6 April 2018]. Available from: https://git-scm.com/doc

JOHN, JULIANNY, MARIA GABRIELA and JEFERSON. Metodología de Desarrollo de Sistemas de Información. *Scribd* [online]. 2 July 2009. [Accessed 2 December 2017]. Available from: https://es.scribd.com/document/365229308/Metodologia-de-Desarrollo-de-Sistemas-de-Informacion-1

CEVALLOS, Karla. METODOLOGÍA DE DESARROLLO ÁGIL. *Ingeniería del Software. Protafolio Digital* [online]. May 2015. [Accessed 2 December 2017]. Available from: https://ingsotfwarekarlacevallos.wordpress.com/category/metodologia-de-desarrollo-agil/

G, Aurelio and ARILLAS. AUP. *Metodología* [online]. 23 July 2017. [Accessed 2 December 2017]. Available from: https://metodologia.es/aup/

SÁNCHEZ, Tamara Rodríguez. *Metodología de desarrollo para la actividad productiva de la UCI*. 21 November 2014.

Data Modeling - UML Diagramming Software. *VP Gallery* [online]. 2010. [Accessed 26 February 2018]. Available from: https://www.visual-paradigm.com/VPGallery/datamodeling/index.html

SOMMERVILLE, Ian. *Software Engineering*. 7th edition. Madrid : Pearson Educación S.A, 2005. ISBN 84-7829-074-5.

UML Modeling - Unified Modeling Language Tool. *VP Gallery* [online]. 2010. [Accessed 26 February 2018]. Available from: https://www.visual-paradigm.com/VPGallery/diagrams/index.html

REYNOSO, Carlos Billy. *Introducción a la Arquitectura de Software* [online]. 15 June 2004. Available from: http://sunshine.prod.uci.cu/book/4e73a4c305717427e9000003/

BASS, L., CLEMENTS, P. and KAZMAN, R. *Software architecture in practice, SEI Series in Software Engineering.* Second Edition. Addison Wesley, 2003.

MOSS, HOLOVATY, Adrian and KAPLAN, Jacob. *Django Book*. Jeremy Hunck, 2008.

CAMACHO, Erika, CARDESO, Fabio and NUÑEZ, Gabriel. *Arquitecturas de software. Guia de estudio.* 2004.

Arquitectura Basada en Componentes. [online]. [Accessed 7 April 2018]. Available from: https://es.scribd.com/doc/14704374/Arquitectura-Basada-en-Componentes

GUERRERO, Carlos A., SUÁREZ, Johanna M. and GUTIÉRREZ, Luz E. Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. *Información tecnológica*. 2013. Vol. 24, no. 3, p. 103–114. DOI 10.4067/S0718-07642013000300012.

TEDESCHI, Nicolás. ¿Qué es un Patrón de Diseño? [online]. 2018. [Accessed 8 April 2018]. Available from: https://msdn.microsoft.com/es-es/library/bb972240.aspx

LARMAN, Craig. *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2da Edición. [no date]. ISBN 970-17-0261-1.

MANUEL ARIAS CALLEJA. *Estándares de codificación* [online]. 2008. Available from: https://es.scribd.com/document/55695081/Estandares-de-codificacion

PORTO, Julián Pérez and GARDEY, Ana. Definición de validación - Qué es, Significado y Concepto. [online]. 2013. [Accessed 9 April 2018]. Available from: https://definicion.de/validacion/

PRESSMAN, Roger S. *Software Engineering. A practitioner´s approach*. 7th edition. 2010.

JOSKOWICK, José. *Reglas y Prácticas en eXtreme Programming*. España : Universidad de Vigo, 2008.

BIBLIOGRAFÍA

¿Que es bootstrap ? Bootstrap es un framework front end, 2015. [online]. [Accessed 4 December 2017]. Available from: http://www.negocioscaninos.com/que-es-bootstrap-bootstrap-framework-front-end/

ALFONSO, J Minguillón, [no date]. *Introducción al Lenguaje de Modelado Unificado* [online]. FUOC. Available from: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/9121/1/Intro_UML.pdf

AM, Vipul and SONPATKI, Prathamesh, 2016. *ReaactJS by Example - Building Modern Web Applications with React* [online]. Packt Publishing Birmingham. ISBN 978-1-78528-964-4. Available from: http://droppdf.com/

ANGULAR, 2018. Angular (@angular) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/angular

Arquitectura Basada en Componentes, [no date]. [online]. [Accessed 7 April 2018]. Available from: https://es.scribd.com/doc/14704374/Arquitectura-Basada-en-Componentes

AURELIA, 2018. Aurelia (@AureliaEffect) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/aureliaeffect

AZAÑA, Daniel López, 2014. ¿Qué es la arquitectura web? [online]. 17 September 2014. [Accessed 4 December 2017]. Available from: http://www.daniloaz.com/es/que-es-la-arquitectura-web/

Backbone.js, el framework para construir aplicaciones usando Javascript siguiendo el patrón MVC alcanza la versión 1.0, [no date]. [online]. [Accessed 4 December 2017]. Available from: https://www.genbetadev.com/desarrollo-web/backbone-js-el-framework-para-construir-aplicaciones-usando-javascript-siguiendo-el-patron-mvc-alcanza-la-version-1-0

BASS, L., CLEMENTS, P. and KAZMAN, R., 2003. *Software architecture in practice, SEI Series in Software Engineering.* Second Edition. Addison Wesley.

BHAUMIK, Snig, 2015. *Bootstrap Essentials. Use the powerful features of bootstrap to*

create responsive and appealing web pages. [online]. Packt Publishing Birmingham. ISBN 978-1-78439-517-9. Available from: http://droppdf.com/

Bootstrap 3 - Genbeta Dev, [no date]. [online]. [Accessed 25 January 2018]. Available from: https://www.genbetadev.com/tag/bootstrap-3

Bootstrap 3 Vs Bootstrap 4 : What's New?, 2017. *BootstrapDash* [online]. [Accessed 22 February 2018]. Available from: https://www.bootstrapdash.com/bootstrap-3-vs-4/

CAMACHO, Erika, CARDESO, Fabio and NUÑEZ, Gabriel, 2004. *Arquitecturas de software. Guia de estudio.*

Casos de Prueba, [no date]. [online]. [Accessed 10 April 2018]. Available from: https://es.scribd.com/document/367743970/Casos-de-Prueba

Centro de Telemática (TLM) | Universidad de las Ciencias Informáticas, 2018. [online]. [Accessed 6 February 2018]. Available from: http://www.uci.cu/investigacion-y-desarrollo/centros-de-desarrollo/centro-de-telematica-tlm

CEVALLOS, Karla, 2015. METODOLOGÍA DE DESARROLLO ÁGIL. *Ingeniería del Software. Protafolio Digital* [online]. May 2015. [Accessed 2 December 2017]. Available from: https://ingsotfwarekarlacevallos.wordpress.com/category/metodologia-de-desarrollo-agil/

Conceptos generales de la arquitectura de aplicaciones web., [no date]. [online]. Available from: http://www.ra-ma.es

CORNEJO, Jose Enrrique González, 2008. El Lenguaje de Modelado Unificado (UML). [online]. January 2008. [Accessed 15 December 2017]. Available from: http://www.docirs.com/uml.htm

CRAVENS, Jesse and BRADY, Thomas Q, 2014. *Building Web Apps with Ember.js. Write ambitious JavaScript* [online]. O'Reilly Media, Inc. ISBN 978-1-4493-7092-3. Available from: http://oreilly.com/catalog/errata.csp?isbn=9781449370923

DARWIN, Peter Bacon and KOZLOWSKI, Pawel, 2013. *Mastering Web Application Development with AngularJS*. Packt Publishing.

Data Modeling - UML Diagramming Software, 2010. *VP Gallery* [online]. [Accessed 26 February 2018]. Available from: https://www.visual-paradigm.com/VPGallery/datamodeling/index.html

DECONCEPTOS.COM, 2017. Concepto de restricción - Definición en DeConceptos.com. [online]. 2017. [Accessed 10 April 2018]. Available from: https://deconceptos.com/general/restriccion

Desarrollador/a de front-end / Angular JS - Globalstudio - Madrid, España (29/01/16) | Domestika, [no date]. [online]. [Accessed 23 October 2017]. Available from: https://www.domestika.org/es/jobs/37504-desarrollador-a-de-front-end-angular-js-madrid-espana

Desarrollo de Software – Metodología Tradicional o Agil ? | PMQuality, [no date]. [online]. [Accessed 2 December 2017]. Available from: https://pmqlinkedin.wordpress.com/about/metodologia-tradicional-o-agil/

DESARROLLOWEB.COM, [no date]. Qué es BackboneJS. *DesarrolloWeb.com* [online]. [Accessed 4 December 2017]. Available from: http://www.desarrolloweb.com/articulos/que-es-backbonejs.html

DÍAZ LAZO, Juliet, PÉREZ GUTIÉRREZ, Adriana and FLORIDO BACALLAO, René, 2011. IMPACTO DE LAS TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES (TIC) PARA DISMINUIR LA BRECHA DIGITAL EN LA SOCIEDAD ACTUAL. *Cultivos Tropicales*. March 2011. Vol. 32, no. 1, p. 81–90.

Difference between the Bootstrap 2 and Bootstrap 3, 2015. *Technotipz* [online]. [Accessed 22 February 2018]. Available from: https://www.technotipz.com/technotipz/difference-bootstrap-2-bootstrap-3/

Documentation for Visual Studio Code, 2018. *Visual Studio Code* [online]. [Accessed 21 March 2018]. Available from: https://code.visualstudio.com/doc

EISENMAN, Bonnie, 2017. *Learning React Native: Building Native Mobile Apps with JavaScript* [online]. O'Reilly Media, Inc. ISBN 978-1-4919-8911-1. Available from: https://books.google.com.cu/books?id=wgg7DwAAQBAJGoogle-Books-ID: wgg7DwAAQBAJ

EMBERJS, 2018. EmberJS (@emberjs) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/emberjs

¿Es Angular 2, Angular 4 o simplemente Angular?, [no date]. [online]. [Accessed 23 November 2017]. Available from: https://www.campusmvp.es/recursos/post/es-angular-2-angular-4-o-simplemente-angular.aspx

Features - PyCharm, 2018. *Python IDE for Professional Developers* [online]. [Accessed 15 December 2017]. Available from: https://www.jetbrains.com/pycharm/features/

G, Aurelio and ARILLAS, 2017. AUP. *Metodología* [online]. 23 July 2017. [Accessed 2 December 2017]. Available from: https://metodologia.es/aup/

GACKENHEIMER, Cory, 2015. *Introduction to React. Using creact to build scalable and efficient user interfaces.* [online]. Editorial Board. ISBN 978-1-4842-1245-5. Available from: https://books.google.com.cu/books?id=NZCKCgAAQBAJGoogle-Books-ID: NZCKCgAAQBAJ

GANDHAM, Mani, 2016. 14 Answers - What are the pros and cons of using Bootstrap in web development? [online]. 2016. [Accessed 4 December 2017]. Available from: https://www.quora.com/What-are-the-pros-and-cons-of-using-Bootstrap-in-web-development

GARRETT, Jesse James, 2015. Ajax: A New Approach to Web Applications | Adaptive Path. [online]. 18 February 2015. [Accessed 21 March 2018]. Available from: http://adaptivepath.org/ideas/ajax-new-approach-web-applications/

Git version control, 2018. *Git* [online]. [Accessed 6 April 2018]. Available from: https://git-scm.com/doc

GONZÁLEZ, Daniel and MARCOS, Mari-Carmen, 2013. Responsive web design: diseño multidispositivo para mejorar la experiencia de usuario. *BiD: textos universitaris de biblioteconomia i documentació* [online]. 2013. Available from: http://bid.ub.edu/es/31/gonzalez2.htm

GONZÁLEZ, Sayda Coello and LEÓN, Rolando Alfredo Hernández, 2012. *El proceso de investigación científica.* 2 edición. Editorial Universitaria.

GOOGLE, 2018. Google Trends. *Google Trends* [online]. 2018. [Accessed 30 January 2018]. Available from: https://trends.google.es/trends/explore?date=today%205-y&geo=ES&q=angular,react,vue.js,ember.js,aurelia Demanda de los frameworks Angular, React, VueJS, EmberJS, Aurelia

GREEN, Brad and SESHADRI, Shyam, 2013. *AngularJS.* O'Reilly Media, Inc.

GUERRERO, Carlos A., SUÁREZ, Johanna M. and GUTIÉRREZ, Luz E., 2013. Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. *Información tecnológica*. 2013. Vol. 24, no. 3, p. 103–114. DOI 10.4067/S0718-07642013000300012.

GUILBAULT, Manuel, 2016. *Learning Aurelia. Harness the power of the next-generation JavaScript Framework, Aurelia, and start creating apps that really set you apart.* [online]. Packt Publishing Birmingham. ISBN 978-1-78588-967-7. Available from: http://droppdf.com/

HOLOVATY, Adrian, MOSSS, Jacob Kaplan and M, Saul Garcia, 2015. *La guía definitiva de Django. Desarrolla aplicaciones Web de forma rápida y sencilla.* [online]. Django Software Corporation. ISBN MA 02111-1307. Available from: http://github.com/saulgm/djangobook.com

IBÁÑEZ, Carlos García, BALLESTEROS, Raqueñ Hervás and GERVÁS, Pablo, 2004. Una arquitectura software para el desarrollo de aplicaciones de generación de lenguaje natural. [online]. 2004. Vol. 33. Available from: http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/3065

INSTITUTO TECNOLÓGICO DE MATEHUALA, 2015. 2.1 Arquitectura de las aplicaciones

Web | Programación Web. [online]. 2015. [Accessed 4 December 2017]. Available from: https://programacionwebisc.wordpress.com/2-1-arquitectura-de-las-aplicaciones-web/

INTERNATION DESIGN FOUNDATION, 2018. Efficiency. *The Glossary of human Computer Interaction* [online]. 2002 2018. [Accessed 29 May 2018]. Available from: https://www.interaction-design.org/literature/book/the-glossary-of-human-computer-interaction/efficiency

Introduction — Vue.js, 2018. *oficial* [online]. [Accessed 21 January 2018]. Available from: https://vuejs.org/v2/guide/Vue.js - The Progressive JavaScript Framework

JIAS, [no date]. Web: ¿Qué es el Framework Bootstrap? Ventajas y Desventajas. – Apuntes de Programación. [online]. [Accessed 4 December 2017]. Available from: http://programacion.jias.es/2015/05/web-%c2%bfque-es-el-framework-bootstrap-ventajas-desventajas/

JOHN, JULIANNY, MARIA GABRIELA and JEFERSON, 2009. Metodología de Desarrollo de Sistemas de Información. *Scribd* [online]. 2 July 2009. [Accessed 2 December 2017]. Available from: https://es.scribd.com/document/365229308/Metodologia-de-Desarrollo-de-Sistemas-de-Informacion-1

KURNIAWAN, Agus, 2017. *Vue.js Programming by Example* [online]. Available from: https://books.google.es/books?id=JDw8DwAAQBAJGoogle-Books-ID: JDw8DwAAQBAJ

LARMAN, Craig, [no date]. *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2da Edición. ISBN 970-17-0261-1.

Las 5 principales ventajas de usar Angular para crear aplicaciones web, [no date]. [online]. [Accessed 3 December 2017]. Available from: https://www.campusmvp.es/recursos/post/las-5-principales-ventajas-de-usar-angular-para-crear-aplicaciones-web.aspx

MANUEL ARIAS CALLEJA, 2008. *Estándares de codificación* [online]. Available from: https://es.scribd.com/document/55695081/Estandares-de-codificacion

MARQUINA, Ernesto and PARRA, Jose David, 2008. *Guía de Patrones, Prácticas y*

Arquitecturas .NET. 2008.

Metodologias de Desarrollo Agil, [no date]. [online]. [Accessed 2 December 2017]. Available from: https://es.scribd.com/document/62533867/Metodologias-de-Desarrollo-Agil

Metodologías tradicionales y metodologías ágiles, [no date]. [online]. [Accessed 2 December 2017]. Available from: http://www.eumed.net/libros-gratis/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm

MOSS, HOLOVATY, Adrian and KAPLAN, Jacob, 2008. *Django Book*. Jeremy Hunck.

NERI, Carlos, 2007. No todo es click. Usabilidad, accesibilidad y experiencia del usuario en la Web. *Scribd* [online]. 2007. [Accessed 9 April 2018]. Available from: https://es.scribd.com/document/51645567/el-concepto-de-usabilidad

NILESH, Jain, PRIYANKA, Mangal and DEEPAK, Mehta, 2014. AngularJS: A Modern MVC Framework in JavaScript. [online]. 2014. Vol. 5, no. 12. Available from: https://pdfs.semanticscholar.org/a8bd/ab90c7b568755d718f9fd0a92221488f757d.pdf

ORDOÑES, Yoanni, AVILÉS, Ernesto, HERNÁNDEZ, Julio and RODRÍGUEZ, Odaysa, 2014. *GRHS: Gestor de Recursos de Hardware y Software.*

ORTÍ, Consuelo Belloch, [no date]. *Las Tecnologías de la Información y las Comunicaciones (TIC)*. Unidad de Tecnología Educativa

OTTO, Mark, THORNTON, Jacob and BOOTSTRAP CONTRIBUTORS, [no date]. Bootstrap. [online]. [Accessed 25 January 2018]. Available from: https://getbootstrap.com/

OXFORD DICTIONARIES DEFINITION, 2018. desig. [online]. 2018. [Accessed 29 May 2018]. Available from: https://en.oxforddictionaries.com/definition/design

OXFORD DICTIONARIES, 2018. restriction. [online]. 2018. [Accessed 29 May 2018]. Available from: https://en.oxforddictionaries.com/definition/restriction

PÉREZ, Javier Eguíluz, 2008. *Introducción a AJAX* [online]. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/introduccion_ajax.pdf

PÉREZ, Javier Eguíluz, 2009a. *Introducción a JavaScript* [online]. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/introduccion_javascript.pdf

PÉREZ, Javier Eguíluz, 2009b. *CSS avanzado* [online]. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/css_avanzado.pdf

Por qué AngularJS, [no date]. [online]. [Accessed 23 October 2017]. Available from: https://desarrolloweb.com/articulos/por-que-angularjs.html

PORTO, Julián Pérez and GARDEY, Ana, 2012. Definición de eficiencia - Qué es, Significado y Concepto. [online]. 2012. [Accessed 9 April 2018]. Available from: https://definicion.de/eficiencia/

PORTO, Julián Pérez and GARDEY, Ana, 2013. Definición de validación - Qué es, Significado y Concepto. [online]. 2013. [Accessed 9 April 2018]. Available from: https://definicion.de/validacion/

PORTO, Julián Pérez and GARDEY, Ana, 2015. Definición de restricción — Definicion.de. *Definición.de* [online]. 2015. [Accessed 10 April 2018]. Available from: https://definicion.de/restriccion/

PORTO, Julián Pérez and MERINO, María, 2012. Definición de diseño — Definicion.de. *Definición.de* [online]. 2012. [Accessed 10 April 2018]. Available from: https://definicion.de/diseno/

PORTO, Julián Pérez and MERINO, María, 2013. Definición de usabilidad - Qué es, Significado y Concepto. [online]. 2013. [Accessed 9 April 2018]. Available from: https://definicion.de/usabilidad/

PostgreSQL: Advantages, [no date]. [online]. [Accessed 15 December 2017]. Available from: https://www.postgresql.org/about/advantages/

PRESSMAN, Roger S, 2010. *Software Engineering. A practitioner´s approach*. 7th edition.

Productos | Universidad de las Ciencias Informáticas, 2018. [online]. [Accessed 6 February 2018]. Available from: http://www.uci.cu/investigacion-y-

desarrollo/productos

PURI, Suchit, 2015. *Ember.js Web Development with Ember CLI* [online]. Packt Publishing Birmingham. ISBN 978-1-78439-584-1. Available from: http://droppdf.com/

¿Qué es Diseño? - Su Definición, Concepto y Significado, 2015. [online]. [Accessed 10 April 2018]. Available from: http://conceptodefinicion.de/diseno/

RAE- ASALE, 2018a. restricción. *Diccionario de la lengua española - Edición del Tricentenario* [online]. 2018. [Accessed 29 May 2018]. Available from: http://dle.rae.es/?id=WEgzpII

RAE- ASALE, 2018b. eficiencia. *Diccionario de la lengua española - Edición del Tricentenario* [online]. 2018. [Accessed 29 May 2018]. Available from: http://dle.rae.es/?id=EPVwpUD

RAE- ASALE, 2018c. diseño. *Diccionario de la lengua española - Edición del Tricentenario* [online]. 2018. [Accessed 29 May 2018]. Available from: http://dle.rae.es/?id=DuKP0H9

React - A JavaScript library for building user interfaces, [no date]. [online]. [Accessed 18 January 2018]. Available from: https://reactjs.org/index.htmlA JavaScript library for building user interfaces

React For Beginners — The best way to learn React, [no date]. [online]. [Accessed 18 January 2018]. Available from: https://reactforbeginners.com/

React integration for ASP.NET MVC | ReactJS.NET, [no date]. [online]. [Accessed 18 January 2018]. Available from: http://reactjs.net/index.html.NET integration for ReactJS

REACT, 2018. React (@reactjs) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/reactjs

REYNOSO, Carlos Billy, 2004. *Introducción a la Arquitectura de Software* [online]. 15 June 2004. Available from: http://sunshine.prod.uci.cu/book/4e73a4c305717427e9000003/

ROMERO, Hermes, 02:59:12 UTC. Metodologias de desarrollo. [online]. 02:59:12 UTC.

[Accessed 2 December 2017]. Available from: https://es.slideshare.net/MeneRomero/metodologias-de-desarrollo

ROSAS, Jesus, 2015. Que es Backbone.js y como funciona. *Azul Web* [online]. 27 November 2015. [Accessed 4 December 2017]. Available from: https://www.azulweb.net/que-es-backbone-js-y-como-funciona/Backbone.js es una librería que utiliza el patrón de arquitectura MVC y que trabaja del lado del Front-end, en este articulo aprenderás como funciona

SAMPIERI, 2006. *Metodología de la Investigación* [online]. Available from: http://sunshine.prod.uci.cu/book/4f2bfb920571740d090000b3/

SÁNCHEZ, Tamara Rodríguez, 2014. *Metodología de desarrollo para la actividad productiva de la UCI*. 21 November 2014.

SANDERS, william, 2013. *Learning PHP design patterns* [online]. O'Reilly Media, Inc. ISBN 978-1-4493-4491-7. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/learning_php_design_patterns.pdf

SHACKEL, Brian and RICHARDSON, Simon, 1991. *Human Factors for Informatics Usability* [online]. New York: Cambridge University Press. ISBN 0 521 36570 8. Available from: https://books.google.com.cu/books?hl=es&lr=&id=KSHrPgLlMJIC&oi=fnd&pg=PA21&dq=definition+of+usability&ots=IWPtOZ-SB9&sig=-TRQ_17PCizZLGV5qIPfMX5E-Xk&redir_esc=y#v=onepage&q&f=false

SIERRA, F., ACOSTA, J., ARIZA, J. and SALAS, M., 2013. Estudio y analisis de los framework en php basados en el modelo vista controlador para el desarrollo de software orientado a la web. . 15 October 2013.

Significado de Diseño, [no date]. *Significados* [online]. [Accessed 10 April 2018]. Available from: http://www.significados.com/diseno/Qué es el Diseño. Concepto y Significado de Diseño: El diseño es constituido por trazos o una delineación con el fin de proyectar un objeto u obra....

SIGNIFICADOS.COM, 2018. Significado de Eficiencia - Qué es, Concepto y Definición. [online]. 2013 2018. [Accessed 9 April 2018]. Available from:

https://www.significados.com/eficiencia/

SOMMERVILLE, Ian, 2005. *Software Engineering*. 7th edition. Madrid: Pearson Educación S.A. ISBN 84-7829-074-5.

TEDESCHI, Nicolás, 2018. ¿Qué es un Patrón de Diseño? [online]. 2018. [Accessed 8 April 2018]. Available from: https://msdn.microsoft.com/es-es/library/bb972240.aspx

The Agile Unified Process (AUP) Home Page, [no date]. [online]. [Accessed 2 December 2017]. Available from: http://www.ambysoft.com/unifiedprocess/agileUP.html

UML Modeling - Unified Modeling Language Tool, 2010. *VP Gallery* [online]. [Accessed 26 February 2018]. Available from: https://www.visual-paradigm.com/VPGallery/diagrams/index.html

VILÁ, Fermí, 2001. *JavaScript Manual* [online]. Available from: http://sunshine.prod.uci.cu/gridfs/sunshine/books/Javascript.pdf

Visual Paradigm 8.0 (formerly VP-UML 8.0) Released, 2010. *Visual Paradigm* [online]. [Accessed 15 December 2017]. Available from: https://www.visual-paradigm.com/aboutus/newsreleases/vpuml80.jsp

Vue.js, [no date]. [online]. [Accessed 21 January 2018]. Available from: https://vuejs.org/

VUE.JS, 2018. Vue.js (@vuejs) | Twitter. [online]. 2018. [Accessed 30 May 2018]. Available from: https://twitter.com/vuejs

What is PostgreSQL, 2018. *PostgreSQL tutorial* [online]. [Accessed 15 December 2017]. Available from: http://www.postgresqltutorial.com/what-is-postgresql/

What is Python? Executive Summary, 2018. *Python.org* [online]. [Accessed 12 December 2017]. Available from: https://www.python.org/doc/essays/blurb/

What Is React? | SpringerLink, [no date]. [online]. [Accessed 18 January 2018]. Available from: https://link.springer.com/chapter/10.1007/978-1-4842-1245-5_1

ANEXOS

ANEXO 1: DICCIONARIO DE DATOS

Tabla 21: Descripción de la entidad _ Grupos

Descripción	Realiza la gestión de grupos del sistema.					
Atributos						
Nombre	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clases válidas	**Clases no válidas**
nombre	Nombre del grupo establecido en el sistema.	cadena de caracteres	no	si	Toma valor cadena de caracteres.	Toma valor cadena de caracteres alfanuméricos y caracteres especiales

Tabla 22: Descripción de la entidad _ Permisos de los grupos

Descripción	Realiza la gestión de los permisos de grupos del sistema.					
Atributos						
Nombre	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clases válidas	**Clases no válidas**
nombre de menú	Nombre del menú establecido en el sistema.	cadena de caracteres	no	si	Toma valor cadena de caracteres.	Toma valor cadena de caracteres alfanuméricos y caracteres especiales
nombre de permiso	Nombre de los permisos establecido en el sistema.	cadena de caracteres	no	si	Toma valor cadena de caracteres.	Toma valor cadena de caracteres alfanuméricos y caracteres especiales

ANEXO 2: ESPECIFICACIONES DEL REQUISITO _ GESTIONAR USUARIO

Tabla 23: Especificación del requisito Gestionar usuario -> Modificar usuario

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar usuario	
Flujo básico Modificar usuario	
1.	Seleccionar usuario
2.	Mostrar formulario con los datos del usuario
3.	Llenar campo de texto a modificar

4.	Seleccionar botón “Aceptar”
5.	Enviar valores
Pos-condiciones	
1.	Actualizar valores en la base de datos
Flujos alternativos	
Flujo alternativo 4.a Campos vacíos	
1.	Señalar campos vacíos
2.	Mostrar notificación: Por favor inserte un valor para este campo.
3.	Llenar campos de texto vacíos
4.	Volver al paso 4 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 4.b Valores incorrectos	
1.	Señalar campos con valores incorrectos
2.	Mostrar notificación referente a las clases válidas para cada campo
3.	Llenar los campos de textos incorrectos
4.	Volver al paso 4 del flujo básico
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios

Conceptos	**Usuarios**	Visibles a la interfaz: Usuario, contraseña, confirmar contraseña, nombre, apellidos, dirección de correo, activo, superusuario, token, grupo, dominio
Requisitos especiales	Usabilidad, Operabilidad	
Asuntos pendientes		
Prototipo elemental de interfaz gráfica de usuario		

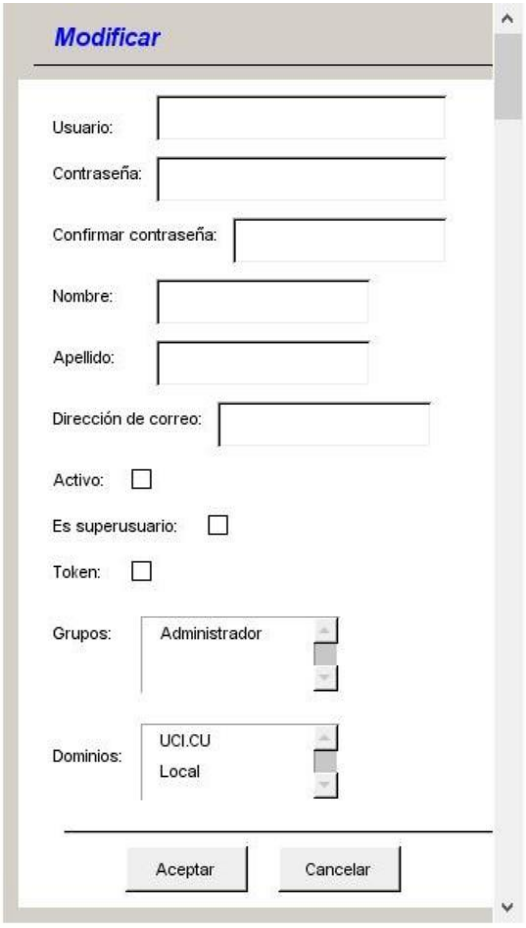


Tabla 24: Especificación del requisito Gestionar usuario -> Eliminar usuario

Precondiciones		*El cliente ha sido validado.*
Flujo de eventos Gestionar usuario		
Flujo básico Eliminar usuario		
1.	Seleccionar usuario	
2.	Seleccionar botón “Eliminar”	
3.	Mostrar notificación: ¿Estás seguro de que quieres eliminar permanentemente el(los) elemento(s) seleccionado(s)?	
4.	Seleccionar botón “Aceptar”	
5.	Enviar valores	
6.	Mostrar notificación: La operación se realizó con éxito.	
Pos-condiciones		
1.	*Eliminar los valores de la base de datos*.	
Validaciones		
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios	
Conceptos	**Usuarios**	*Visibles a la interfaz:* *Usuarios*

Requisitos especiales	Usabilidad, Operabilidad
Asuntos pendientes	
Prototipo elemental de interfaz gráfica de usuario	

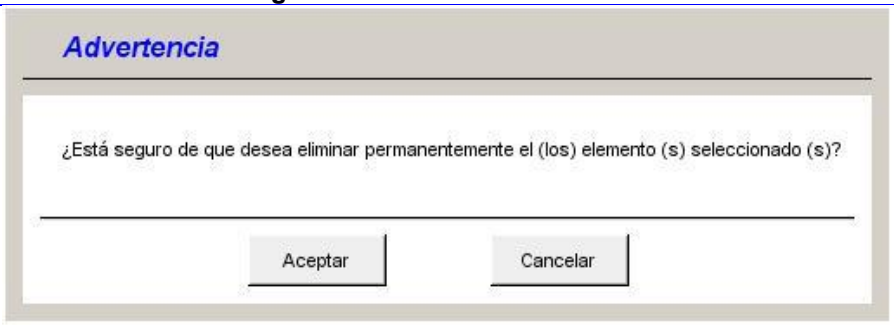


Tabla 25: Especificación del requisito Gestionar usuario -> Mostrar detalles de usuario

Precondiciones	*El cliente ha sido validado.*	
Flujo de eventos Gestionar usuario		
Flujo básico Mostrar detalles de usuario		
1.	Seleccionar usuario	
2.	Seleccionar botón “Mostrar detalles”	
Pos-condiciones		
1.	Mostrar los valores de la base de datos correspondientes al usuario seleccionado (usuario, nombre, apellido, dirección de correo electrónico, activo, superusuario, grupo, dominio)	
Validaciones		
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios	
Conceptos	**Usuarios**	Visibles a la interfaz: Usuario, nombre, apellidos, dirección de correo, activo, superusuario, token, grupo, dominio
Requisitos especiales	Funcionabilidad, Precisión.	
Asuntos pendientes		
Prototipo elemental de interfaz gráfica de usuario		

Tabla 26: Especificación del requisito Gestionar usuario -> Cambiar contraseña

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar usuario	
Flujo básico Cambiar contraseña	
1.	Seleccionar usuario
2.	Seleccionar botón “Cambiar contraseña”
3.	Llenar el campo de texto contraseña
4.	Llenar el campo de texto confirmar contraseña
5.	Seleccionar botón “Aceptar”
6.	Enviar valores
Pos-condiciones	
1.	Actualizar valores en la base de datos
Flujos alternativos	
Flujo alternativo 5.a Campos vacíos	
1.	Señalar campos vacíos
2.	Mostrar notificación: Por favor inserte un valor para este campo.
3.	Llenar campos de texto vacíos
4.	Volver al paso 5 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 5.b Contraseña incorrecta	
1.	Señalar el campo confirmar contraseña
2.	Mostrar notificación: La confirmación no tiene el mismo valor que la contraseña.

3.	Actualizar el campo de texto confirmar contraseña
4.	Volver al paso 5 del flujo básico
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios

Conceptos	**Usuarios**	Visibles a la interfaz: contraseña, confirmar contraseña
Requisitos especiales	Usabilidad, Operabilidad	
Asuntos pendientes		

Prototipo elemental de interfaz gráfica de usuario

Tabla 27: Especificación del requisito Gestionar usuario -> Listar usuario

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar usuario	
Flujo básico Listar usuario	
1.	Listar usuarios existentes en la base de datos
2.	Mostrar de todos los usuarios el valor usuario
3.	Mostrar de todos los usuarios el valor nombre
4.	Mostrar de todos los usuarios el valor apellido
5.	Mostrar de todos los usuarios el valor ultimo inicio de sección
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios

Conceptos	**Usuarios**	*Visibles a la interfaz:* *Usuario.*
Requisitos especiales	Funcionabilidad, Precisión.	
Asuntos pendientes		
Prototipo elemental de interfaz gráfica de usuario		

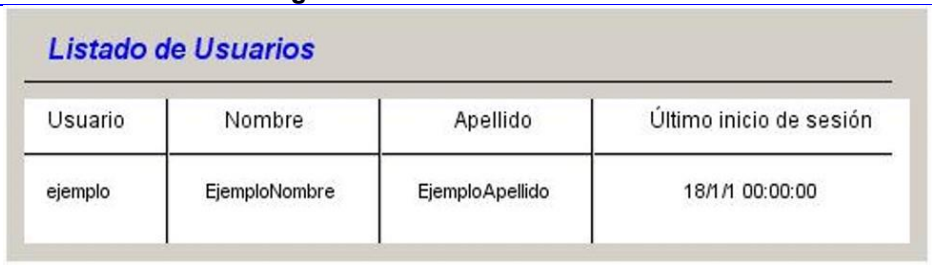
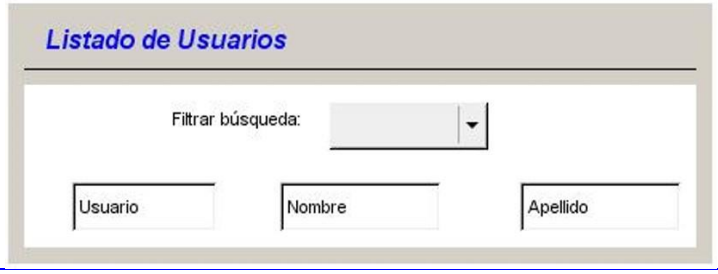


Tabla 28: Especificación del requisito Gestionar usuario -> Buscar usuario

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar usuario	
Flujo básico Buscar usuario	
1.	Llenar el campo de texto del buscador con los datos usuario o nombre o apellido
2.	Seleccionar botón “Buscar”
3.	Enviar valores
Pos-condiciones	
1.	Mostrar valores de la base de datos
Flujos alternativos	
Flujo alternativo 2.a No existen el usuario	
1.	No se muestra información
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 2.b Valores incorrectos	
1.	Mostrar notificación referente a las clases válidas para cada campo
2.	Llenar los campos de textos incorrectos
3.	Volver al paso 2 del flujo básico
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 2: Descripción de la entidad _ Usuarios

Conceptos	**Usuarios**	*Visibles a la interfaz:* *Usuarios.*
Requisitos especiales	Funcionabilidad, Precisión.	
Asuntos pendientes		

Prototipo elemental de interfaz gráfica de usuario



ANEXO 3: ESPECIFICACIONES DEL REQUISITO _ GESTIONAR GRUPO

Tabla 29: Especificación del requisito Gestionar grupo -> Modificar grupo

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar grupo	
Flujo básico Modificar grupo	
1.	Seleccionar grupo
2.	Mostrar formulario con los datos del grupo
3.	Llenar campo de texto a modificar
4.	Seleccionar botón “Aceptar”
5.	Enviar valores
Pos-condiciones	
1.	Actualizar valores en la base de datos
Flujos alternativos	
Flujo alternativo 4.a Campos vacíos	
1.	Señalar campo vacío
2.	Mostrar notificación: Por favor inserte un valor para este campo.
3.	Llenar campo de texto vacío
4.	Volver al paso 4 del flujo básico
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 4.b Valores incorrectos	
1.	Señalar campo con valor incorrecto
2.	Mostrar notificación referente a la clase válida para el campo
3.	Llenar campo de texto incorrecto
4.	Volver al paso 4 del flujo básico
Pos-condiciones	

1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 21: Descripción de la entidad _ Grupos.

Conceptos	**Grupos**	*Visibles a la interfaz:* *Nombre*
Requisitos especiales	Usabilidad, Operabilidad	
Asuntos pendientes		

Prototipo elemental de interfaz gráfica de usuario

Tabla 30: Especificación del requisito Gestionar grupo -> Eliminar grupo

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Gestionar grupo	
Flujo básico Eliminar grupo	
1.	Seleccionar grupo
2.	Seleccionar botón “Eliminar”
3.	Mostrar notificación: ¿Estás seguro de que quieres eliminar permanentemente el(los) elemento(s) seleccionado(s)?
4.	Seleccionar botón “Aceptar”
5.	Enviar valores
6.	Mostrar notificación: La operación se realizó con éxito.
Pos-condiciones	
1.	Eliminar los valores de la base de datos.
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0*y* Tabla 21: Descripción de la entidad _ Grupos.

Conceptos	**Grupos**	*Visibles a la interfaz:* *Grupos*
Requisitos especiales	Usabilidad, Operabilidad	

Asuntos pendientes	

Prototipo elemental de interfaz gráfica de usuario

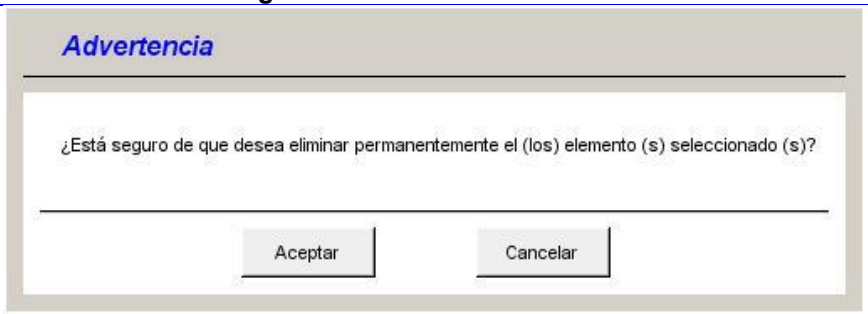


Tabla 31: Especificación del requisito Gestionar grupo -> Listar grupo

Precondiciones		*El cliente ha sido validado.*
Flujo de eventos Gestionar grupo		
Flujo básico Listar grupo		
1.	Listar grupos existentes en la base de datos	
2.	Mostrar de todos los grupos el valor nombre	
Pos-condiciones		
1.	N/A	
Validaciones		
1.	Ver Figura 3: Modelo conceptual de Xilema base Web 2.0 y Tabla 21: Descripción de la entidad _ Grupos .	
Conceptos	**Grupos**	*Visibles a la interfaz:* *Grupos*
Requisitos especiales	Funcionabilidad, Precisión.	
Asuntos pendientes		

Prototipo elemental de interfaz gráfica de usuario

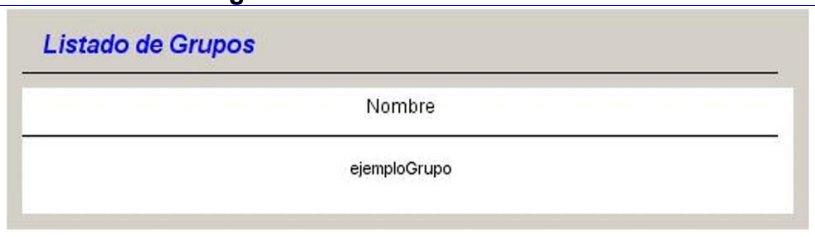


Tabla 32: Especificación del requisito Gestionar grupo -> Buscar grupo

Precondiciones		*El cliente ha sido validado.*
Flujo de eventos Gestionar grupo		
Flujo básico Buscar grupo		
1.	Llenar el campo de texto del buscador con el dato nombre	
2.	Seleccionar botón “Buscar”	
3.	Enviar valores	

Pos-condiciones	
1.	Mostrar valores de la base de datos
Flujos alternativos	
Flujo alternativo 2.a No existen el grupo	
1.	No se muestra información
Pos-condiciones	
1.	N/A
Flujos alternativos	
Flujo alternativo 2.b Valores incorrectos	
1.	Mostrar notificación referente a las clases válidas para cada campo
2.	Llenar los campos de textos incorrectos
3.	Volver al paso 2 del flujo básico
Pos-condiciones	
1.	N/A
Validaciones	
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 21: Descripción de la entidad _ Grupos.

Conceptos	**Grupos**	*Visibles a la interfaz:* *Grupos*
Requisitos especiales	Funcionabilidad, Precisión.	
Asuntos pendientes		

Prototipo elemental de interfaz gráfica de usuario

ANEXO 4: ESPECIFICACIONES DEL REQUISITO _ PERMISOS DE LOS GRUPOS

Tabla 33: Especificación del requisito Permisos de los grupos -> Buscar permisos

Precondiciones	*El cliente ha sido validado.*
Flujo de eventos Permisos de los grupos	
Flujo básico Buscar permisos	
1.	Llenar el campo de texto del buscador con los datos nombre de menú o nombre de permiso
2.	Seleccionar botón “Buscar”
3.	Enviar valores
Pos-condiciones	
1.	Mostrar valores de la base de datos
Flujos alternativos	

Flujo alternativo 2.a No existen el permiso		
1.	Mostrar notificación	
Pos-condiciones		
1.	N/A	
Flujos alternativos		
Flujo alternativo 2.b Valores incorrectos		
1.	Mostrar notificación	
2.	Mostrar el campo de texto vacío	
Pos-condiciones		
1.	N/A	
Validaciones		
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 22: Descripción de la entidad _ Permisos de los grupos.	
Conceptos	**Permisos de Grupos**	Visibles a la interfaz: Permisos
Requisitos especiales	Funcionabilidad, Precisión.	
Asuntos pendientes		
Prototipo elemental de interfaz gráfica de usuario		

Tabla 34: Especificación del requisito Permisos de los grupos -> Seleccionar grupo

Precondiciones	*El cliente ha sido validado.*	
Flujo de eventos Permisos de los grupos		
Flujo básico Seleccionar grupo		
1.	Desplegar botón “Grupo”	
2.	Seleccionar grupo por el dato nombre	
Pos-condiciones		
1.	Marcar los permisos del grupo seleccionado	
Validaciones		
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0 *y* Tabla 22: Descripción de la entidad _ Permisos de los grupos.	
Conceptos	**Permisos de Grupos**	*Visibles a la interfaz:* *Grupos*

Requisitos especiales	Funcionabilidad, Precisión.
Asuntos pendientes	

Prototipo elemental de interfaz gráfica de usuario

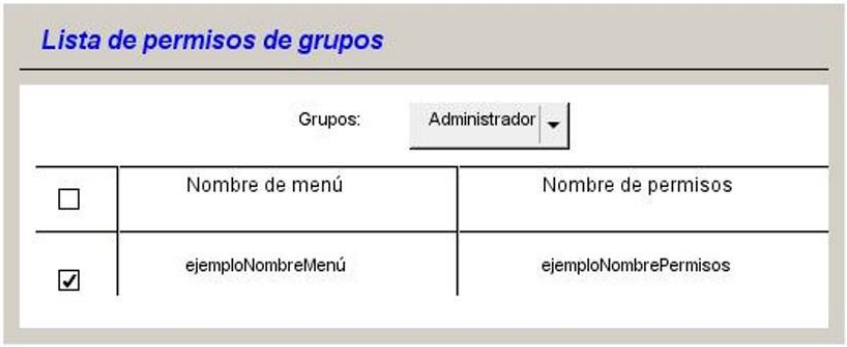
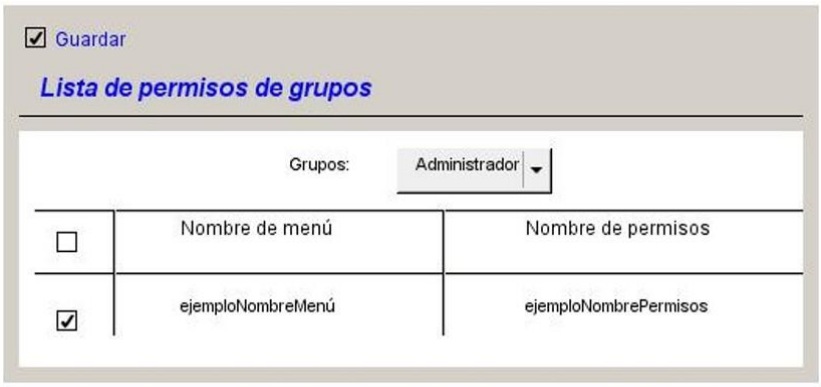


Tabla 35: Especificación del requisito Permisos de los grupos -> Guardar permisos

Precondiciones	*El cliente ha sido validado.*	
Flujo de eventos Permisos de los grupos		
Flujo básico Guardar permisos		
1.	Marcar permisos	
2.	Seleccionar botón “Guardar”	
Pos-condiciones		
1.	Guarda los valores en la base de datos	
Flujos alternativos		
Flujo alternativo 2.a No se realizaron cambios		
1.	Mostrar notificación	
Pos-condiciones		
1.	N/A	
Validaciones		
1.	*Ver* Figura 3: Modelo conceptual de Xilema base Web 2.0*y* Tabla 22: Descripción de la entidad _ Permisos de los grupos*.*	
Conceptos	**Permisos de Grupos**	*Visibles a la interfaz:* *Permisos*
Requisitos especiales	Funcionabilidad, Precisión.	
Asuntos pendientes		

Prototipo elemental de interfaz gráfica de usuario



ANEXO 5: DESCRIPCIONES DE CLASES

Tabla 36: Descripción de la clase _ Dominio

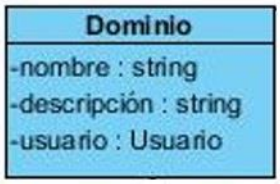
Número del módulo	1
Número de la clase	2
Clase	Dominio
Propósito	Contiene diferentes tipos de dominio según la necesidad de la empresa.
Descripción	
Observaciones	

Tabla 37: Descripción de la clase _ Modelo

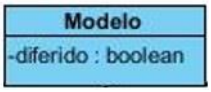
Número del módulo	1
Número de la clase	3
Clase	Modelo
Propósito	Clase definida por el marco de trabajo Django.
Descripción	
Observaciones	

Tabla 38: Descripción de la clase _ Menú

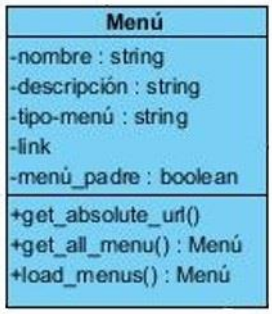
Número del módulo	1
Número de la clase	4
Clase	Menú
Propósito	Contiene la configuración general para todos lo menús y submenús que necesite el sistema.
Descripción	
Observaciones	

Tabla 39: Descripción de la clase _ Sistema

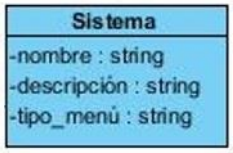
Número del módulo	1
Número de la clase	5
Clase	Sistema
Propósito	Es la clase referente al Módulo Sistema, el cual es el menú principal que contiene toda la gestión de usuarios, grupos y permisos.
Descripción	
Observaciones	

Tabla 40: Descripción de la clase _ Usuario

Número del módulo	1
Número de la clase	6
Clase	Usuario

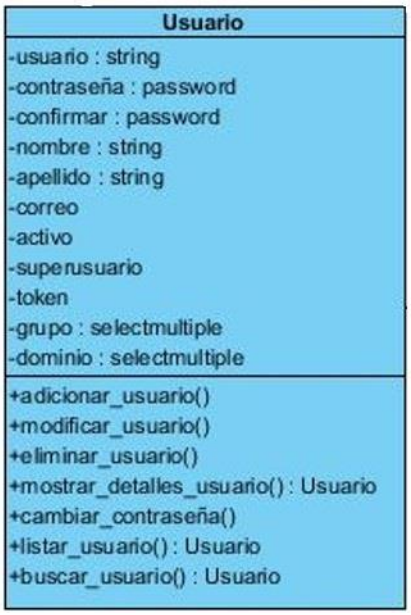
Propósito	Gestionar toda la información referente a los usuarios existentes en el sistema.
Descripción	
Observaciones	

Tabla 41: Descripción de la clase _ Grupo

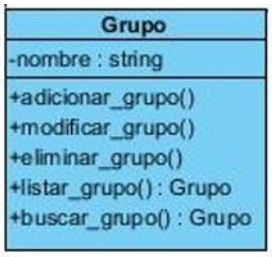
Número del módulo	1
Número de la clase	7
Clase	Grupo
Propósito	Gestionar toda la información referente a los grupos existentes en el sistema.
Descripción	
Observaciones	

Tabla 42: Descripción de la clase _ Permisos menú

Número del módulo	1

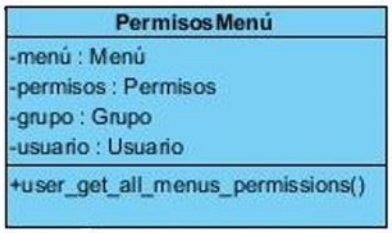
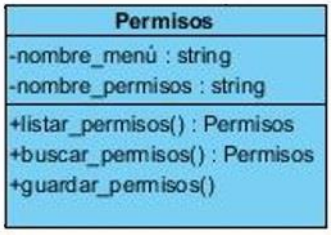
Número de la clase	8
Clase	Permisos Menú
Propósito	Contiene la relación entre los permisos y los usuarios y grupos correspondientes a los cuales se les ha otorgado dichos permisos.
Descripción	
Observaciones	

Tabla 43: Descripción de la clase _ Permisos

Número del módulo	1
Número de la clase	9
Clase	Permisos
Propósito	Gestionar toda la información referente a los permisos de los grupos del sistema.
Descripción	
Observaciones	

ANEXO 6: DESCRIPCIONES DE CASOS DE PRUEBAS

Descripción general: Gestionar usuario

Gestionar usuario es un macroproceso que lleva en sí otros requisitos tales como: adicionar usuario, modificar usuario, eliminar usuario, mostrar detalles de usuario, cambiar contraseña, listar usuarios y buscar usuario.

Condición de ejecución.

El cliente ha sido validado

Tabla 44: Caso de prueba _ Gestionar usuario (primera parte)

Escenario	Descripción	Variable usuario	Variable contraseña	Variable confirmar	Variable nombre	Variable apellido
Adicionar	Adicionar usuarios al sistema.	V wendy	V WTD	V WTD	V wendy	V trujillo
		I w3ndy	V WTD	V WTD	V wendy	V trujillo
		V wendy	V WTD	I wen	V wendy	V trujillo
		V wendy	V WTD	V WTD	V wendy	V trujillo
		I	V WTD	V WTD	I	I
Modificar	Modificar los usuarios del sistema.	V snayder	V SJZC	V SJZC	V snayder	V zarate
		I Sna6der	V SJZC	V SJZC	V snayder	V zarate
		V snayder	V SJZC	I snay	V snayder	V zarate
		V snayder	V SJZC	V SJZC	V snayder	V zarate
		I	V SJZC	V SJZC	I	I
Cambiar contraseña	Cambiar la contraseña de un usuario del sistema.	-	V wtrujillo	V wtrujillo	-	-
		-	V wtrujillo	I delgado	-	-
		-	I	V wtrujillo	-	-
Buscar	Buscar usuarios en el sistema.	V wendy	-	-	I Wen	I capote
		V wendy	-	-	V	I capote
		V	-	-	V wendy	V trujillo
		I wen	-	-	V wendy	V
		I wen	-	-	I Wen	V trujillo

Tabla 45: Caso de prueba _ Gestionar usuario (segunda parte)

Variable dirección de correo	Variable activo	Variable super usuario	Variable token	Variable grupo	Variable dominio	Respuesta del sistema	Flujo central

V wendy@uci.cu	V ☑	V ☑	V ☑	V administrador	V local	El usuario es adicionado al sistema	Dar clic en el menú superior en Sistema. Dar clic en la opción Usuarios en el menú izquierdo. Seleccionar el ícono Adicionar. Llenar campo de texto usuario, contraseña, confirmar, nombre, apellido, dirección de correo, activo, superusuario, token. Seleccionar el grupo, dominio. Seleccionar botón “Aceptar”.
V wendy@uci.cu	V ☒	V ☒	V ☒	V administrador	V local	Muestra notificación de valor inválido.	
V wendy@uci.cu	V ☑	V ☒	V ☑	V administrador	V local	Muestra notificación de valor inválido.	
I Wendy*uci.cu	V ☒	V ☑	V ☒	V administrador	V local	Muestra notificación de valor inválido..	
I	V ☑	V ☑	V ☒	V administrador	V local	Muestra notificación de campo vacío.	
V snayder@uci.cu	V ☑	V ☑	V ☑	V administrador	V local	El usuario es modificado en el sistema	Dar clic en el menú superior en Sistema. Dar clic en la opción Usuarios en el menú izquierdo. Seleccionar el ícono Modificar. Llenar campo de texto usuario, contraseña, confirmar, nombre, apellido, dirección de correo, activo,
V snayder@uci.cu	V ☒	V ☒	V ☒	V administrador	V local	Muestra notificación de valor inválido.	
V snayder@uci.cu	V ☒	V ☑	V ☒	V administrador	V local	Muestra notificación de valor inválido.	
I Snayder*uci.cu	V ☑	V ☒	V ☑	V administrador	V local	Muestra notificación de valor inválido..	
I	V ☑	V ☒	V ☒	V administrador	V local	Muestra notificación de campo vacío.	

							superusuario, token. Seleccionar el grupo, dominio. Seleccionar botón “Aceptar”.
-	-	-	-	-	-	La contraseña ha sido cambiada en el sistema.	Dar clic en el menú superior en Sistema. Dar clic en la opción Usuarios en el menú izquierdo. Seleccionar el ícono Cambiar contraseña Llenar campo de texto contraseña, confirmar. Seleccionar botón “Aceptar”.
-	-	-	-	-	-	Muestra notificación de valor inválido.	
-	-	-	-	-	-	Muestra notificación de campo vacío.	
-	-	-	-	-	-	Muestra notificación de valor inválido.	Dar clic en el menú superior en Sistema. Dar clic en la opción Usuarios en el menú izquierdo. Seleccionar opción Buscar. Llenar campo de texto usuario, nombre, apellido. Seleccionar botón “Buscar”.
-	-	-	-	-	-	Muestra notificación de campo vacío.	
-	-	-	-	-	-	Muestra notificación de campo vacío.	
-	-	-	-	-	-	Muestra notificación de campo vacío.	
-	-	-	-	-	-	Muestra notificación de valor inválido.	

Descripción general: Gestionar grupo

Gestionar grupo es un macroproceso que lleva en sí otros requisitos tales como: adicionar grupo, modificar grupo, eliminar grupo, listar grupos y buscar grupo.

Condición de ejecución.

El cliente ha sido validado

Tabla 46: Caso de prueba _ Gestionar grupo

Escenario	Descripción	Variable nombre	Respuesta del sistema	Flujo central
Adicionar	Adicionar grupos al sistema.	V trabajador	El grupo es adicionado al sistema	Dar clic en el menú superior en Sistema. Dar clic en la opción Grupos en el menú izquierdo. Seleccionar el ícono Adicionar. Llenar campo de texto nombre. Seleccionar botón "Aceptar".
		I trab!ado$	Muestra notificación de valor inválido.	
		I	Muestra notificación de campo vacío.	
Modificar	Modificar los grupos del sistema.	V estudiante	El grupo es modificado en el sistema	Dar clic en el menú superior en Sistema. Dar clic en la opción Grupos en el menú izquierdo. Seleccionar el ícono Modificar. Llenar campo de texto nombre. Seleccionar botón "Aceptar".
		I estud1ant3s	Muestra notificación de valor inválido.	
		I	Muestra notificación de campo vacío.	
Buscar	Buscar grupos en el sistema.	V trabajador	El grupo es mostrado en el sistema.	Dar clic en el menú superior en Sistema. Dar clic en la opción Grupos en el menú izquierdo. Seleccionar opción Buscar. Llenar campo de texto nombre. Seleccionar botón "Buscar".
		I estud1ant3s	Muestra notificación de valor inválido.	
		I	Muestra notificación de campo vacío.	

Descripción general: Permisos de los grupos

Permisos de los grupos es un macroproceso que lleva en sí otros requisitos tales como: listar permisos, buscar permisos, seleccionar grupo y guardar permisos.

Condición de ejecución.

El cliente ha sido validado

Tabla 47: Caso de prueba _ Permisos de los grupos

Escenario	Descripción	Variable nombre de menú	Variable nombre de permisos	Respuesta del sistema	Flujo central
Buscar	Buscar permisos en el sistema.	V Usuarios	V Cambiar Contraseña	El permiso es mostrado en el sistema.	Dar clic en el menú superior en Sistema. Dar clic en la opción Permisos de Grupo en el menú izquierdo. Seleccionar opción Buscar Llenar campo de texto nombre de menú, nombre de permisos. Seleccionar botón “Buscar”.
		V	V Cambiar Contraseña	El permiso es mostrado en el sistema.	
		V Usuarios	V	El permiso es mostrado en el sistema.	
		V Usuarios	I Camb!ar Contras3ña	Muestra notificación de valor inválido.	
		V Usu$rios	V Cambiar Contraseña	Muestra notificación de valor inválido.	

ANEXO: ENTREVISTA

Universidad de la Ciencias Informáticas	
Centro de desarrollo de Telemática	
Nombre: Yoanny Torres Rubio	**Fecha:** octubre/2017
Objetivo: Obtener información sobre el estado actual del software Xilema base Web, los problemas que presenta.	

Preguntas:

1. ¿Qué patrón de diseño presenta el software Xilema base Web?

_____

2. ¿Qué problemas presenta en la actualidad el software Xilema base Web?

_____

3. ¿Existe documentación asociada al software Xilema base Web?

Si _____ No _____

a) Existe documentación relacionada específicamente con la configuración de la interfaz?

Bastante _____ No mucha _____ Ninguna _____

4. ¿Presenta un diseño adaptativo el software Xilema base Web?

Si _____ No _____ No sé _____

5. ¿Utiliza tecnología Ajax para la comunicación entre el cliente y el servidor?

Si _____ No _____ No sé _____