Tecnologico Nacional De México

INSTITUTO TECNOLOGICO DE APIZACO

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

CONACYT
Consejo Nacional de Ciencia y Tecnología

“Instituto Tecnológico De Apizaco”

División De Estudios De Posgrado e Investigación.

T e s i s:

“Diseño e implementación de una base de datos relacional utilizando ORM Programing Technique, Spring Security e Hibernate para un Head End System.”

QUE PARA OPTAR POR EL GRADO DE:

Maestro En Sistemas Computacionales.

Presenta:

Ing. Luis Angel Espina Hernández.

Directores de tesis:

Mtro. Higinio Nava Bautista.

M en C. José Juan Hernández Mora.

Fecha de Graduación:

12 de Diciembre de 2018.

TECNM
TECNOLOGICO NACIONAL DE MEXICO

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Apizaco

Apizaco, Tlax., 18 de Septiembre de 2018

ASUNTO: **Aprobación del trabajo de Tesis de Maestría.**

DR. JOSÉ FEDERICO CASCO VÁSQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN.
P R E S E N T E.

Por este medio se le informa a usted, que los integrantes de la **Comisión Revisora** para el trabajo de tesis de maestría que presenta el **ING. LUIS ANGEL ESPINA HERNÁNDEZ** con N° de control **M10370700,** candidato al grado de **Maestro en Sistemas Computacionales** y egresado del **Instituto Tecnológico de Apizaco,** cuyo tema es **"DISEÑO E IMPLEMENTACIÓN DE UNA BASE DE DATOS RELACIONAL UTILIZANDO ORM PROGRAMING TECHNIQUE, SPRING SECURITY E HIBERNATE PARA UN HEAD END SYSTEM",** fue:

A P R O B A D O

Lo anterior, al valorar el trabajo profesional presentado por el candidato y constatar que las observaciones que con anterioridad se le marcaron así como correcciones sugeridas para su mejora ya han sido realizadas.

Por lo que se avala se continúe con los trámites pertinentes para su titulación.

Sin otro particular por el momento, le envió un cordial saludo.

LA COMISIÓN REVISORA

M.D.S. HIGINIO NAVA BAUTISTA

M.C. JOSÉ JUAN HERNÁNDEZ MORA

M.C. MARÍA GUADALUPE MEDINA BARRERA

M.C. MARÍA JANAÍ SÁNCHEZ HERNÁNDEZ

C. p.- Interesado.

Instituto Tecnológico de Apizaco
Carretera Apizaco-Tzompantepec, Esq. con Av. Instituto Tecnológico S/N
Conurbado Apizaco-Tzompantepec, Tlaxcala, Méx.
C.P. 90300, Apizaco, Tlax. Tels. 01241 4172010, Ext. 146, 246
e-mail: depi@apizaco.tecnm.mx, **www.itapizaco.edu.mx**

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Apizaco

Apizaco, Tlax., 15 de Octubre de 2018

No. de Oficio: DEPI/519/18

ASUNTO: **Se Autoriza Impresión de Tesis de Grado.**

ING. LUIS ANGEL ESPINA HERNÁNDEZ
CANDIDATO AL GRADO DE MAESTRO
EN SISTEMAS COMPUTACIONALES
No. de Control: **M10370700**
P R E S E N T E.

Por este medio me permito informar a usted, que por aprobación de la Comisión Revisora asignada para valorar el trabajo, mediante la Opción: **I Tesis de Grado por Proyecto de Investigación**, de la **Maestría en Sistemas Computacionales**, que presenta con el tema: "**DISEÑO E IMPLEMENTACIÓN DE UNA BASE DE DATOS RELACIONAL UTILIZANDO ORM PROGRAMING TECHNIQUE, SPRING SECURITY E HIBERNATE PARA UN HEAD END SYSTEM**" y conforme a lo establecido en el Procedimiento para la Obtención del Grado de Maestría en el Instituto Tecnológico, la División de Estudios de Posgrado e Investigación a mi cargo le emite la:

AUTORIZACIÓN DE IMPRESIÓN

Debiendo entregar un ejemplar del mismo debidamente encuadernado y seis copias en CD en formato PDF, para presentar su Acto de Recepción Profesional a la brevedad.

Sin otro particular por el momento, le envío un cordial saludo.

A T E N T A M E N T E
EXCELENCIA EN EDUCACIÓN TECNOLÓGICA®
PENSAR PARA SERVIR, SERVIR PARA TRIUNFAR®

DR. JOSÉ FEDERICO CASCO VÁSQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN.

SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
DIVISIÓN DE ESTUDIO DE POSGRADO E INVESTIGACIÓN

JFCV/MAEH*mebr.
C.p. Expediente.

Instituto Tecnológico de Apizaco
Carretera Apizaco-Tzompantepec, Esq. con Av. Instituto Tecnológico S/N
Conurbado Apizaco-Tzompantepec, Tlaxcala, Méx.
C.P. 90300, Apizaco, Tlax. Tels. 01241 4172010, Ext. 146, 246
e-mail: depi@apizaco.tecnm.mx, www.apizaco.tecnm.mx

Agradecimiento.

A:

Mis padres, mis hermanas, tíos, primos, amigos y compañeros, que siempre estuvieron a mi lado para darme una mano y así salir adelante, por todo el apoyo recibido por parte de todas las personas allegadas a mí, de manera especial al M.S.C Agustín Sánchez Atonal por todos los consejos recibidos durante mi estancia en Querétaro, a la Lic. Doris Castro por ser una muy buena maestra.

Al maestro Higinio Nava Bautista mi profesor y asesor de tesis, por siempre brindar su conocimiento y siempre impulsarme a seguir adelante durante la elaboración de tesis.

Al comité José Juan Hernandez Mora, Guadalupe Medina Barrera, María Janai Sánchez Hernandez por sus valiosas aportaciones para mejorar la presente tesis.

A todos los profesores del Instituto Tecnológico de Apizaco y a todo el personal de la empresa EOSTech.

Agradezco al Instituto Tecnológico de Apizaco y al Consejo Nacional de Ciencia y Tecnología (Conacyt) por el apoyo económico brindado a través de los diversos programas de becas.

A mis amigos de carrera Carlos Raúl por ser como mi hermano y ser el mejor amigo que e tenido en años, Mary Carmen que a pesar de todos los momentos difíciles siempre estuvo para brindar su mano como apoyo, Moisés por brindarme todos sus conocimientos informáticos en momentos que lo requerí, Luis Alberto por ser un excelente amigo y a Yoselim por brindarme su más sincera amistad y por todos los momentos compartidos durante la carrera, siempre estarán en mis recuerdos.

De igual manera agradezco a Karina Morales por ser una excelente amiga y consejera y todas las palabras de aliento brindadas y todas las tardes de su compañía, a Lizeth Rivera por mostrarme que la inocencia y rectitud son virtudes muy importantes, a Lizbeth Zamora por ser una excelente persona, por brindarme todo su apoyo, cariño, confianza, tiempo, sus palabras de aliento y por estar ahí para brindarme una mano y un abrazo cuando más los necesite, de igual manera y de manera muy especial agradecer a Valeria Jara por mostrarme el otro lado de la vida y que no necesitas nada más que ser tú mismo para ser feliz , no tengo palabras para agradecer, por todo el cariño, tiempo y el apoyo brindado, por mostrarme que la amistad y la confianza son un tesoro muy valioso.

A todos aquellos que han hecho posible que yo siga aquí, muchas gracias.

Dedicatoria.

Para las únicas personas que han estado cerca de mí en todo momento, desde el día de mi nacimiento hasta este mismo instante. Siempre listos para regalarme exactamente lo que necesito: una sonrisa, un consejo, un abrazo o hasta un regaño.

Absolutamente todo en esta vida se lo debo a ustedes. Todo lo que tengo, todo lo que soy incluso todo lo que algún día pueda llegar a obtener, es gracias a ustedes, Mi madre y su ternura, mi padre y su sacrificio, son una gran motivación y mi principal motivo para siempre intentar ser una mejor persona.

Seguiré su ejemplo y sus enseñanzas todos los días de mi vida.

¡¡¡Gracias padres míos!!! Gracias por todo.

Declaración de autenticidad.

Por la presente declaro que, salvo cuando se haga referencia específica al trabajo de otras personas, el contenido de esta tesis es original y no se ha presentado total o parcialmente para su consideración para cualquier otro título o grado en esta o cualquier otra Universidad. Esta tesis es resultado de mi propio trabajo y no incluye nada que sea el resultado de algún trabajo realizado en colaboración, salvo que se indique específicamente en el texto.

Luis Angel Espina Hernández.

Resumen

La presente tesis describe el desarrollo y la implementación de una base de datos, así como la implementación de Spring Security en un sistema informático HES (Head End System), construido con la metodología de desarrollo ágil SCRUM y programación en bloques que fueron utilizadas por el equipo de desarrollo que se formó durante el periodo de estancias realizadas en la empresa EOSTech SA de CV, en esta empresa se facilitaron los documentos y el equipo necesario para realizar el presente trabajo.

Las aplicaciones de medición inteligente son herramientas que permiten analizar y gestionar sistemas de gran tamaño desde un punto central de control. Actualmente, existe una tendencia mundial para adaptar nuevas tecnologías en una innumerable cantidad de aplicaciones debido a los múltiples beneficios que ofrecen hoy en día. Tomando en cuenta esta situación en la presente tesis se habla del diseño de una base de datos relacional para un sistema HES el cual se va a comunicar con un sistema MDM (Meter Data Management) y una SON (Smart Object Network) todos ellos planeados para un sistema AMI (Advanced Metering Infrastructure). En primera instancia se expone el estado del arte de los dispositivos inteligentes, sistemas AMI , de las bases de datos relacionales y de seguridad de Spring Security, así como el lugar que ocupa en el sistema HES. Posteriormente propone un prototipo del sistema, que consta de, medidores inteligentes, un concentrador de datos, la lógica de negocios y una interfaz gráfica para el control del sistema.

El trabajo realizado se presenta como solución en la adquisición de datos para el control del consumo de energía eléctrica en todos los sectores, el sistema obtiene los datos de los medidores y los transmite a través de un módulo de comunicación denominado red de objetos inteligentes, este tipo de red se caracteriza por estar formada por dispositivos programados para funcionar de manera independiente, minimizando la interacción con recursos humanos, el hecho de incursionar nuevos modelos de gestión y automatización de los servicios prestados despierta el interés de la producción nacional de buscar alternativas para integrar y adaptar tecnologías a las necesidades de la industria eléctrica.

El aporte de este proyecto es el diseño y el desarrollo de una base de datos relacional con JPA (Java Persistence API), Hibernate y Oracle, utilizando la arquitectura MVC (Modelo-Vista-Controlador) trabajando en conjunto con la tesis propuesta por el ingeniero Moises Morales Guzman la cual fue titulada “Diseño e Implementación de una Interfaz Gráfica de Usuario para un Sistema Head End de una Red Distribuida de Medidores Inteligentes” donde se describe la parte del desarrollo de la vista y finalmente en colaboración con la licenciada Doris Castro Velazco en la propuesta de tesis titulada “Desarrollo e implementación de la lógica de control del proceso de desarrollo de un sistema HEAD-END para una red de dispositivos inteligentes” se describe la parte del desarrollo del controlador.

Para el desarrollo de la seguridad se utilizó el framework de Spring Security y Spring MVC trabajando sobre la vista y la base de datos, con usuarios y roles.

Abstract

This thesis describes the development and implementation of a database, as well as the implementation of Spring Security in a computer system HES (Head End system), built with the agile development methodology SCRUM and programming in blocks that were used by the development team that was formed during the period of stays carried out in the company EOSTech SA de CV, the documents and equipment needed to carry out this work were provided in this company.

Smart-metering applications are tools that allow you to analyze and manage large systems from a central control point. Today, there is a worldwide trend to adapt new technologies to countless applications because of the many benefits they offer nowadays. Taking into account this situation in this thesis we talk about the design of a relational database for a HES system which is going to communicate with a system MDM (Meter data Management) and SON (Smart Object Network) all planned for an AMI system (Advanced Metering Infrastructure). In the first instance the state of the art of the intelligent devices, AMI systems, of the relational and security databases of Spring security is exposed, as well as the place that occupies in the HES system. Later it proposes a prototype of the system, consisting of intelligent meters, a data concentrator, the business logic and a graphical interface for the control of the system.

The work is presented as a solution in the acquisition of data for the control of the consumption of electricity in all sectors, the system obtains the data of the meters and transmits them through a communication module called Smart Object Network, this type of network is characterized by being made up of devices programmed to operate independently minimizing the interaction with human resources, the fact of incursing new models of management and automation of the services rendered arouses the interest of national production to look for alternatives to integrate and adapt technologies to the needs of the electrical industry.

The contribution of this project is the design and development of a relational database with JPA (Java Persistence API), Hibernate and Oracle, using the MVC architecture (Model-View-Controller) working in conjunction with the thesis proposed by the engineer Moises Morales Guzman which was titled "Design and Implementation of a graphical user interface for a Head End system of a distributed network of intelligent meters" which describes the part of the development of the vision and finally in collaboration with the graduate Doris Castro Velazco in the thesis proposal entitled "Development and implementation of the logic of control of the development process of a HEAD-END system for a network of Intelligent devices" describes the part of the development of the controller.

For the development of security was used the Framework of spring security and spring MVC working on the view and the database, with users and roles.

Contenido

Índice de figuras

Índice de tablas

Capítulo 1. Introducción.

1.1. - Descripción del problema.

EOSTech una empresa ubicada en el estado de Querétaro que forma parte del grupo tecnologías EOS, la cual se dedicada a desarrollar softwares innovadores, así que se comenzó con un proyecto para la generación de una red inteligente diseñada por comisión federal de electricidad (CFE) la cual se encarga de brindar el servicio a toda la república mexicana.

Tradicionalmente, los proyectos orientados a actualizar la infraestructura del sistema eléctrico se enfocan en la construcción de nuevas plantas generadoras, nuevas líneas de transmisión, sub-estaciones y equipo asociado, sin embargo, paulatinamente se vuelve más costoso y más difícil la obtención de datos (Strzelecki, 2010).

Acorde a la actual tendencia mundial, la respuesta a los problemas que enfrenta el sector eléctrico, radica en la actualización de los sistemas eléctricos, utilizando tecnologías disponibles hoy en día para optimizar su operación y desarrollar un sistema más robusto, flexible, eficiente, observable, controlable y confiable. Los sistemas eléctricos que poseen estas características se denominan: Redes Eléctricas Inteligentes (REI), o bien, por el termino: “Smart Grid” (Strzelecki, 2010).

El principal objetivo de la implementación de la REI es: actualizar y optimizar el sistema de potencia existente. No obstante, para que la REI se vuelva realidad, primero es necesario desplegar un sistema integral de medición y comunicación denominado: AMI.

La AMI al pretender ser un gran sistema, tiene que integrar equipos de medición inteligente (Zong Lin, 2009), redes de comunicación y software para medir, recopilar, almacenar y analizar datos sobre el consumo de energía eléctrica y el estado de la red eléctrica inteligente, además de una comunicación bidireccional con CFE y el servicio y así realizar acciones de supervisión y control a distancia.

El sistema AMI están generalmente integrados, por medidores inteligentes ubicados en el punto de interconexión de cada usuario, un sistema de gestión de datos localizados en las instalaciones de CFE y el sistema HES que comunica ambas partes.

La AMI proporciona sistemas observables y controlables a distancia, haciendo posible la automatización de los mismos. Además, no está limitado a solamente a mediciones eléctricas, sus características permiten cuantificar a cualquier variante que pueda ser digitalizada como: presión, iluminación, temperatura, velocidad del viento, consumo de agua o gas entre otros.

Este trabajo de tesis ofrece inicialmente un panorama general sobre el sistema AMI y analiza su estado del arte y describe brevemente los inicios de estas tecnologías, su gradual desarrollo y las causas que lo motivaron. Con esa información como preámbulo, el resto de la investigación se centra en la creación de la base de datos relacional de un sistema Head End para un AMI. Dichas características en común y son alistadas a continuación.

- Medidor inteligente.
- Centro de gestión AMI.
- Medios de comunicación.
- Módulo de medios de recolección de datos.
- Gabinete de medidores.
- Indicador de consumo (Display MRE).
- Supervisión y control en tiempo real de los dispositivos conectados al sistema.

Para satisfacer los puntos anteriores, el sistema de medición avanzada AMI consta de medidores inteligentes, sistema de red mesh y un sistema de MDM.

El primer dispositivo que encontramos dentro de la red inteligente es el medidor modular (MM), el cual es un dispositivo inteligente el cual se caracteriza por poder enviar un aviso ante cualquier evento o falla en tiempo real. Este medidor realiza el registro recolectando información para la facturación dentro de comisión, todo esto de manera automática.

El nombre de medidor modular viene del hecho de que estos dispositivos serán ubicados dentro de un gabinete el cual solo las personas autorizadas de comisión federal de electricidad tienen acceso. Estos gabinetes son denominados gabinete modular de medidores (GMM), debido a que los Smart meter están conectados dentro de uno, estos tendrán dos salidas de comunicación, una por antena RF y otra por una salida infrarroja.

Los gabinetes modulares deben estar ubicados a un costado de los transformadores o generadores de energía, en lo alto de los postes o en raros casos a un costado del edificio en la parte exterior, todo ello por motivos de seguridad, ya que es la razón primordial para la integración de los nuevos medidores dentro de él y es un esfuerzo para dificultar el acceso a personas ajenas a comisión federal de electricidad.

Otra de las ventajas que ofrece el sistema, es que los usuarios podrán checar su consumo de energía en la pantalla de un MRE (Modulo Remoto de Energía).

Este dispositivo que es entregado por CFE de manera gratuita, una vez dado de alta el servicio, el MRE deberá ser ubicado dentro de cada uno de los hogares, y estará comunicado directamente al medidor correspondiente, esto es posible gracias al uso de la tecnología PLC (Power Line Comunication).

AMR (Automated Meter Reading) basada en PLC es un método donde los datos son transmitidos sobre las líneas de transición eléctrica hacia la subestación, de aquí estos datos son retransmitidos a una computadora central en las oficinas de CFE. Este método podría ser considerado como un tipo de sistema de red fija, la red utilizada es la red MESH(apreciada en la ilustración 1.1). Este sistema principalmente utilizado para las lecturas de energía eléctrica.

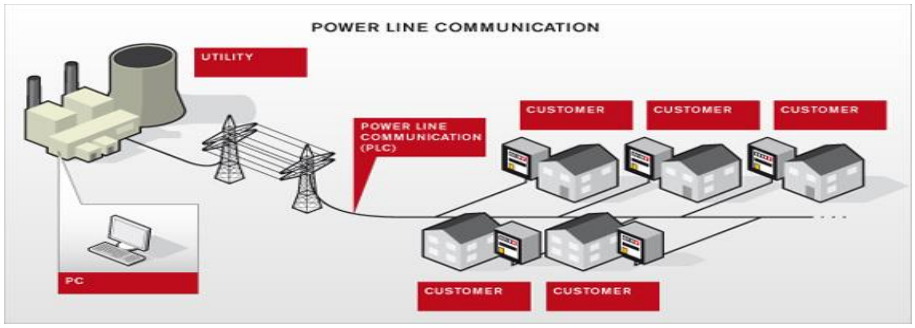


Ilustración 1. 1 Tecnología PLC (Power Line Comunication) (CFE, 2007).

El medidor inteligente cuenta con un radio de corto alcance, esto permite enlazarlos de forma inalámbrica al HES, formando una red mesh de dispositivos bajo el protocolo de red local de radio así como el protocolo TCP/IP, el objetivo es administrar la red local de dispositivos y

fungir una función de enlace entre la red y el sistema HES y viceversa como se aprecia en la ilustración 1.2.

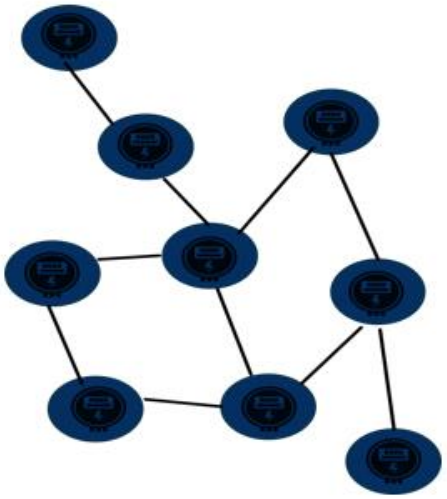


Ilustración 1. 2 Red SON (Elaboración Propia)

Finalmente, el sistema HES al trabajar con un modelo vista controlador incluyo también la implementación de una interfaz web que reside en el sistema. Dicha interfaz permite visualizar la información obtenida por los dispositivos inteligentes y controlar la red desde cualquier computadora que el personal de comisión federal tendrá acceso ya que se les asignara un rol y un usuario con el cual tendrán acceso a los diferentes niveles con los que cuenta la plataforma.

Como se mencionó anteriormente: se envían datos generados por los medidores inteligentes y se envían hacia el Head End System donde además de eso se realizan tareas, como asociación de medidores, de dispositivos, de tarjetas de control, el sistema HES es la comunicación entre la Utility de comisión federal de electricidad y los dispositivos para que pueda realizar todas las operaciones mencionadas, así como la lectura, altas y bajas del servicio, remplazó de equipos ETC.

1.2. - Planteamiento del problema.

Implementar un sistema de recolección de datos llamado Head End System capaz de almacenar información de la red de medidores inteligentes, además de tener comunicación con el módulo MDM, el sistema incluye el diseño de la interfaz gráfica, lógica de negocios y el modelado de la base de datos.

El software debe brindar, interfaz web y comunicación con base de datos para realizar operaciones como: alta de usuarios y medidores, edición, actualización y bajas.

Además de la comunicación con un MDM que es el encargado de guardar toda la información de los medidores que recibe comandos y operaciones que necesitan ser ejecutadas, el cual comunica en tiempo real, las tareas que deben ser asignadas.

La aplicación a desarrollar deberá ejecutar las tareas mencionadas con anterioridad dependiendo de la prioridad que el trabajador asigne, enviando las solicitudes e informando si las tareas fueron exitosas o fracasaron, así que el sistema HES funge como mediador de la red de medidores MESH y el sistema MDM.

La aplicación deberá brindar seguridad a nivel de base de datos tanto como de acceso a la información mostrada en pantalla.

1.3. – Justificación.

Actualmente, las empresas suministradoras de energía eléctrica buscan estrategias rentables para mejorar y eficientar las operaciones del sistema eléctrico (Roadmap, 2012), y los procesos de lecturas implican una serie de gastos para la comisión federal de electricidad. Para cubrir esta oportunidad la empresa EOSTech desarrolla una aplicación de tipo AMI para satisfacer las necesidades.

Considerando todos los beneficios que ofrece un sistema HES y, por otro lado, los altos costos de dichos sistemas de medición y control; en esta tesis se muestra el desarrollo de una base de datos relacional para el sistema HES así como la seguridad entre la base de datos y la interfaz web.

El software a desarrollar deberá ser capaz de intermediar comunicación entre la red Mesh y el MDM, así como recibir las tareas y verificar que sean ejecutadas, mandando señales al MDM si la tarea fue terminada correctamente o la tarea fracaso todo esto se registrara en la base de datos como historial.

1.4. – Objetivos.

1.4.1. – Objetivo general del proyecto.

Desarrollar una base de datos relacional para el sistema Head End System que sea capaz de guardar información de todo tipo de dispositivos inteligentes, utilizando las tecnologías ORM con JPA e Hibérnate, además de robustecer la seguridad con Spring Security en las interfaces web.

1.4.2. - Objetivos específicos.

- Modelar la base de datos en Oracle.
- Diseñar la estructura de la base de datos necesaria para el sistema definiendo sus restricciones y limitaciones.
- Mapeo relacional de la base de datos con Hibernate
- Creación de métodos Getter y Setter de cada enditad de la base de datos.
- Mapeo de entidades con anotaciones JPA.
- Relaciones entre tablas con Hibernate.
- Usuarios Roles y privilegios con Spring Security.
- Seguridad de acceso a la base de datos.

1.5. - Pregunta de investigación.

¿Es posible desarrollar una base de datos utilizando ORM para hacer más eficientes y robustas las transacciones de un Head End System con metodologías de desarrollo ágil?

1.6. – Alcances.

Esta tesis presenta la investigación documental y la creación e implementación de una base de datos para un sistema de medición basado en esquemas que definen una AMI. Contempla la

creación de entidades de la base de datos, la integración con la interfaz gráfica y la lógica de negocios, así como la seguridad en la plataforma con las interfaces web,

Adicional a los elementos mencionados, una AMI normalmente incluye diseños de diversos softwares y sistemas de información, ya sea la comunicación entre los medidores o el MDM para analizar la información recopilada. Esta tesis no cubre estos sistemas informáticos, se enfoca en la creación de la base de datos relacional y presenta además el módulo de seguridad y el control de contenido. En otras palabras, no realiza ningún estudio o análisis con la información obtenida, solo se limita a la base de datos y a la información obtenida.

1.7. - Estado del arte.

La implantación del sistema AMI será el resultado de un proceso evolutivo más que de un hallazgo en particular. Sin embargo, las primeras aplicaciones capaces de guardar información y realizar tareas de manera remota fueron desarrolladas a principios de la década de los setentas, empleando comunicaciones a través de línea telefónica. Dichos experimentos dieron como resultado la patente titulada "Sensor Motoring Device" publicada en 1974 (U.S Patente nº 3842208, 1974) y, posteriormente, en la patente que lleva por título "Apparatus And Method for Remote Sensor Moniroting, Metering And Control" (Estados Unidos Patente nº 4241237, 1984). Esta última describe un dispositivo remoto que inicia una llamada telefónica a una unidad central para reportar la información obtenida en un determinado periodo de tiempo.

Estas primeras investigaciones fueron la base para los primeros sistemas AMR (Automatic Meter Reading) que tuvieron aparición en la década de los ochentas. Tales sistemas permitían establecer una comunicación unidireccional, desde un grupo de medidores hacia un punto central, con la finalidad de reportar información sobre el consumo de los usuarios del servicio eléctrico para fines de facturación.

En la década de los noventas, los avances tecnológicos en el área de telecomunicaciones hicieron factible la evolución de los sistemas AMR hacia los sistemas AMI. La principal diferencia entre ellos radica en que los sistemas AMI utilizan un medio de comunicación bidireccional (envió y recepción de datos), que permite no solamente supervisar los dispositivos en campo, sino también controlarlos.

Años más tarde, en (Wollenberg, 2005) fue introducido el término "Smart Grid" para describir un sistema eléctrico totalmente distribuido, con inteligencia en cada uno de sus componentes, además de un uso extensivo de tecnologías de comunicación y computo. Desde entonces, varias organizaciones, grupos de trabajo, universidades y centros de investigación de todo el mundo, han dedicado parte de sus actividades al estudio y desarrollo de aplicaciones y nuevas tecnologías orientadas a la creación de la REI. Algunas de las organizaciones más destacadas se enlistan a continuación.

a) Instituto de ingenieros en electricidad y electrónica (IEEE)

A la fecha han sido publicados cerca de 2500 artículos de REI en alrededor de 40 revistas de la IEEE. Además, cuenta con alrededor de 100 estándares considerando los 40 ya publicados, que se encuentran en desarrollo y mencionados en el documento "NIST Framework and Roadmap for Smart Grid Interoperability" (IEEE I. S., 2012).

Adicionalmente a la gran cantidad de recursos impresos y digitales, la IEEE representa una comunidad global de expertos en cientos de especialidades quienes respaldan las sociedades de organización del conocimiento, consejos y comités técnicos, grupos de afinidad y redes virtuales. En conjunto, estos organismos ofrecen a la industria una gran cantidad de información, dirección y desarrollo de nuevos productos para el mercado.

b) *European Technology Platform (ETP)*

También llamada "Smart Grids ETP", es el principal foro europeo para la caracterización de políticas de investigación tecnológica para desarrollar los recursos necesarios para la REI. También funge como vínculo entre, las iniciativas estadounidenses y europeas relacionadas con la REI. (Grids, 2012).

La "Smart Grids ETP" se involucra activamente con las partes interesadas (investigadores, académicos, sociedad civil, industria), proyectos de investigación financiados y organizaciones en todo el mundo en un amplio rango de actividades relevantes con la investigación, desarrollo e innovación de las redes eléctricas en Europa (Grids, 2012).

c) *Universidad Carnegie Mellon*

Ubicada en la ciudad de Pittsburg, Pensilvania, es uno de los más destacados centros de investigación superior de los estados unidos en el área de informática y robótica.

Esta universidad será la sede de un nuevo centro de investigaciones de redes eléctricas inteligentes como parte de la asociación industrial-académica de 5 millones de dólares con la "Semiconductor Research Corporation" (SRC). Esta nueva asociación denominada "Energy Research Initiative" (ERI), conjugara a las compañías que se encuentran dentro del mercado energético con investigadores universitarios para intentar resolverla necesidad mundial por fuentes de energía alternativas para la nueva industria emergente. La ERI, gestionada por la SRC, atacara inicialmente dos áreas críticas para una generación eficiente y distribución de fuentes de energía renovables: sistemas fotovoltaicos y tecnologías relacionadas con REI (Mellon, 2012).

El centro de investigaciones eléctricas en la universidad de Carnegie Mellon soportara la incorporación de fuentes de energía renovables y proporcionara herramientas y modelado, simulación y control necesarias para manipular, optimizar y asegurar el sistema de potencia (Mellon, 2012)

d) *Instituto Nacional de Estándares y Tecnología (NIST)*

En los estados unidos de américa, NIST proporciona soporte a uno de los puntos clave en el desarrollo de REI reuniendo a los fabricantes, consumidores, empresas suministradoras y organismos reguladores para desarrollar estándares de interoperabilidad. Desde su creación en 1901, NIST se ha consolidado como un intermediario que trabaja en colaboración con la industria y otras agencias de gobierno. (NIST, 2010)

La misión de NIST ha sido promover la innovación y la competitividad empresarial con el desarrollo de estándares, mediciones y tecnología para promover el desarrollo económico y mejorar la calidad de vida. (NIST, 2010)

La "Ley de independencia y seguridad eléctrica" (ley pública estadounidense 110-140, también conocida como "EISA"), se le otorgo a la NIST, la responsabilidad primaria para coordinar el desarrollo de una estructura que incluya protocolos y estándares base para la gestión de la

información y alcanzar asi la interoperabilidad de los “”dispositivos y sistemas de una REI (NIST, 2010).

e) *“NSF FREEDM Systems Center”*

Localizado es la universidad de carolina del norte en EUA, se encuentra actualmente desarrollando lo que llaman “Transformaciones inteligentes de estado sólido”. Este tipo de transformaciones son más eficientes y adaptables que los trasformadores comúnmente utilizados y representan un gran avance para la REI, permitiendo que el flujo de energía eléctrica sea controlado y redirigido en forma similar a como la información es encaminada a internet. (University, 2011).

El previsto sistema FREEDM es una revolucionaria res eléctrica basada en electrónica de potencia, comunicaciones digitales con alto ancho de banda y control distribuido. Es radicalmente diferente a la red eléctrica actual, ya que reemplaza dispositivos electromagnéticos, como los trasformadores de 60 Hz, con trasformadores de estado sólido. En este sistema, dispositivos de protección de estado sólido también sustituyen interruptores mecánicos. (University, 2011).

El control de flujo de potencia de cuatro cuadrantes que proporciona el transformador de estado sólido permitirá la incursión de generación distribuida a la red, sin efectos adversos para usuarios cercanos, además de proporcionar una inmejorable calidad de energía (University, 2011).

El sistema FREEDM podrá considerarse como el “internet de la energía”, ya que transformará la industria eléctrica así como el internet transformo la industria de la computación; del paradigma de una computadora central, a la computación distribuida que tenemos hoy en día. Un cambio de magnitud será acompañado por una innovación masiva de tecnologías de energía renovable. El sistema FREEDM permitirá a cada usuario tener una participación activa en el consumo y generación de energía. (Systems, 2013).

f) *Universidad de Sídney.*

De forma similar a los casos anteriores, la universidad de Sídney y la empresa Energy Australiana formaron, en 2019 un convenio de colaboración para dirigir el desarrollo de las redes eléctricas inteligentes en Australia y entrenar a la siguiente generación ingenieros de potencia. (Sidney, 2012)

La sociedad pactada, creara un centro de experiencia en la universidad de Sídney para el desarrollo de tecnología, donde los principales objetivos de investigación son fuentes de energía renovables como la solar y eólica, así como electrónica asociada para facilitar la conversión de energía y su integración a la red eléctrica. (Sidney, 2012)

Las bases de datos constituyen una parte fundamental para gestionar los datos en diferentes grados de complejidad con el fin de cumplir en su totalidad con el modelamiento que se realiza en un contexto determinado (Kroenke, 2013).

Surgen entonces varias tecnologías para llevar a cabo dicho modelamiento que facilita un mejor manejo y mayor conocimiento de la forma como se accede a los datos y la manera como se puede mantener las aplicaciones por un determinado tiempo aumentando su calidad (Kroenke, 2013). A continuación se enlistan algunos trabajos utilizando base de datos e Hibernate.

Las instituciones, universidades y empresas privadas mencionadas en esta sección, son una pequeña muestra de los esfuerzos realizados y el interés a nivel mundial por el desarrollo de este tipo de tecnologías.

Documentos y artículos publicados por IEEE (Institute of Electrical and Electronics Engineers), NIST (National Institute of Standards and Technology) y EPRI (Electric Power Research Institute) son la base de la investigación teórica de este trabajo. Por otro lado, el desarrollo de la parte práctica se basó principalmente en los medidores y gabinetes de la empresa EOSTech. Algunos de estos documentos o notas de aplicación se mencionan a continuación.

- En (Kes, 2012) es descrito a grandes rasgos, el diseño del hardware de un medidor eléctrico monofásico. Semana algunos aspectos importantes sobre la calibración de convertidores analógico-digital de aproximaciones sucesivas y el flujo de información apropiado de un medidor eléctrico.

- En (Himani Pandey, 2018) se presenta un sistema de gestión de red basado en la web en el que la plataforma J2EE constituye la base de la implementación. La arquitectura propuesta incluye el entorno técnico de varias tecnologías web, tales como Java, JSP (páginas de servidor Java), CSS (hojas de estilo en cascada), HTML (lenguaje de marcado de hipertexto), servicios web, Ajax, Spring 4.x, Hibernate3.x, JBoss8 (IDE y servidor de aplicaciones) y Oracle 11g. Incorpora servidor Front-End, servicios web Back-End y servidor de base de datos.
- En (Schauer, 2012) se explica el diseño del medidor de energía eléctrica demostrativo, expone algunos detalles importantes sobre la sección de sensores de corriente, plantea algunas ideas para el flujo de información del programa de medición y algunos circuitos de educación para señales de voltaje y corriente.
- En (Wei, 2010) explica cómo opera la escena virtual java3D como un todo en bases de datos relacionales de objetos, se usó un esquema persistente basado en Hibernate para persistir la escena virtual Java3D en Oracle. De acuerdo con la estructura en árbol de la escena virtual Java3D, se crearon 8 clases persistentes y sus asociaciones bidireccionales a través de Hibernate Object / Relational Mapping y mapeos de asociación. Los modelos 3D se almacenaron y modelaron en Oracle 11g Spatial.
- En (Knirsch, 2012) expone detalladamente las ecuaciones que definen las principales variables eléctricas y su respectiva programación en lenguaje C. también expone las ecuaciones comúnmente empleadas para estimas fasores, aplicando la transformada directa de Fourier (DFT) por el método de correlación. Empleando todas estas ecuaciones propone un algoritmo de medición de variables eléctricas que realiza la mayoría de sus cálculos en el dominio de tiempo.
- En (Language, 2015) es un sistema de prueba usa principalmente J2EE como base de desarrollo, usa Eclipse como herramienta de desarrollo, usa Oracle como base de datos, usa Struts, Hibernate y Spring framework para realizar la arquitectura MVC. La sección de análisis de la demanda introduce la demanda global del sistema de examen en línea y los requisitos funcionales del módulo del sistema. La parte del diseño del sistema presenta el diseño del sistema, el diseño de la base de datos y el diseño del módulo del sistema

- En (Harris, 2012) se explica brevemente algunas de las nuevas características que presenta el convertidor analógico-digital de aproximaciones sucesivas de 16 bits, embebiendo el microcontrolador, en comparación con los otros convertidores de 12 bits. Dentro de las principales ventajas, se encuentra la inclusión por hardware que tiende a estabilizar los resultados de las conversiones efectuadas por el dispositivo.
- En (Babu, 2016) menciona que las bases de datos relacionales han sido la opción predominante para back-ends en aplicaciones empresariales durante varias décadas. JDBC, una API de Java, que se utiliza para desarrollar tales aplicaciones y la persistencia de datos en el back-end requiere mucho tiempo y esfuerzo. JDBC hace que la lógica de la aplicación se vincule estrechamente con la base de datos y, por lo tanto, es inadecuada para crear aplicaciones empresariales que deben adoptar requisitos dinámicos el objetivo es desarrollar una aplicación comercial que utilice Hibernate que pueda comunicarse con un back-end asi como con una base de datos.
- En (Single-Phase Electricity Meter, 2013) se expone el diseño de un medidor eléctrico utilizando un microcontrolador de la familia kinetis. Señala también las ventajas que presenta un nuevo núcleo, entre las que destacan una mayor velocidad de procesamiento, nuevos periféricos orientados a mediciones y amplificadores de ganancia programable en las entradas diferentes.
- En (Xia, Yu, & Tang, 2009) describe el enfoque ORM que se realiza por primera vez en Hibernate que es un proyecto de código abierto para sistemas Java, el mapeo relacional de objetos (ORM) en software de computadora es una técnica de programación para convertir datos entre sistemas de tipo incompatible en bases de datos relacionales y lenguajes de programación orientados a objetos.
- La referencia (MQX-Enabled, 2012) expone en detalle, el diseño de un medidor eléctrico demostrativo utilizando un microcontrolador. Explica detalladamente las ventajas que ofrece el microcontrolador mencionado, propone circuitos de educación de señal usando diferentes tipos de sensores y menciona el uso de un módulo de radio que opera con protocolo ZigBee.
- En (Devi & Priya, 2016) describe como crear Business Intelligence Solution para Sriram Industries y Sriram Wire Products, que es la pequeña y la microempresa. El software se

crea utilizando las tecnologías de código abierto como la plataforma J2EE, el marco Spring MVC, Hibernate ORM, el gráfico Jfree y el PDF itext para que sea asequible para las empresas.

- En (Higa, 2013) se plantea un algoritmo de medición de variables eléctricas que trabajan en el dominio de la frecuencia. Para ello, primero se utiliza la FFT para obtener el espectro de frecuencia completo a partir de una señal en el dominio del tiempo.
- En (Sergeev & Matulevicius, 2017) propone un enfoque para el desarrollo de aplicaciones web utilizando la plataforma Spring. La propuesta es respaldada por la herramienta de complemento Eclipse IDE, que reconoce las capturas de configuración de Spring Security, sus anotaciones, y las visualiza en los modelos de control de acceso basado en roles (RBAC). Los modelos RBAC se representan utilizando el lenguaje de modelado SecureUML. El complemento se valida a través de una encuesta realizada por los desarrolladores de software.
- En (Zao, 2012) describe la teoría básica sobre la transformada rápida de Fourier (FFT) y como se implementaría dicha transformación de un dispositivo de 16 procesadores que funcionen en forma paralela.
- En (Meurice, Nagy, & Cleve, 2016) se presenta un enfoque soportado por herramientas, que permite a los desarrolladores analizar cómo el código fuente y el esquema de base de datos co-evolucionaron en el pasado y simular un cambio de esquema de base de datos y determinar automáticamente el conjunto de código fuente lugares que se verían afectados por este cambio. El enfoque ha sido diseñado para tratar con sistemas Java que usan marcos dinámicos de acceso a datos como JDBC, Hibernate y JPA.
- En (Inga, 2012) se describe un procesamiento de auto calibración para el convertidor digital de aproximaciones sucesivas, embebido en el microcontrolador. Además introduce un nuevo periférico, también embebido denominado Bloque de Retardos Programables (PBD) que proporciona un control total sobre los tiempos de disparo del convertidor.
- En (Węgrzynowicz, 2013) se presentan las negatividades de ORM Hibernate Estos antipatrones se centran en los problemas del lado propietario de las colecciones, los tipos y anotaciones de Java utilizados en las asignaciones, así como en el procesamiento de

las colecciones. Cada anti patrón consiste en la descripción de un problema junto con un código de muestra, las consecuencias negativas del rendimiento y la solución recomendada. El rendimiento se analiza en términos del número y la complejidad de la declaración de base de datos emitida, Las muestras de código ilustran cómo los anti patronos disminuyen el rendimiento y cómo implementar las asignaciones para acelerar los tiempos de ejecución.

- En (Luis Puebla, 2013) se documenta un controlador para el uso de paneles líquidos (LSD) empleando diferentes microcontroladores dentro del que se encuentra el MCF51EM256. En otras palabras, se describe la arquitectura de un programa de abstracción de hardware para el uso inmediato de este tipo de paneles LSD.
- En (Bak, Sakowicz, & Napieralski, 2006) es una guía para el desarrollo rápido de aplicaciones J2EE y su implementación en un contenedor liviano usando una aplicación de ejemplo. El alcance de la base de datos con la integración de los siguientes marcos: Spring Inversion of Control, Hibernate, Struts y Acegi Security. También se presentan herramientas adicionales como las herramientas XDoclet e Hibernate para acelerar el proceso de desarrollo.
- Y finalmente en (Shi & Xu, 2016) proponen un modelo mejorado de control de acceso basado en roles para recursos categorizados, en combinación con requisitos especiales de sistemas comunitarios inteligentes. Diseñan un modelo de control de acceso basado en roles para recursos categorizados, que integra la información de categoría de la comunidad en la definición de roles para limitar el número de roles. El nuevo modelo se implementó completamente en un sistema de gestión comunitaria con 14500 usuarios de 14 comunidades. Se compara este sistema con Spring Security, un marco de seguridad de código abierto existente y demostramos las ventajas de nuestro modelo de control de acceso.

Capítulo 2. Marco teórico.

2.1. - Introducción.

A consecuencia del desarrollo tecnológico alcanzado en la actualidad, se vuelve cada vez más común escuchar acerca de dispositivos o tecnologías inteligentes en todas las áreas de la industria; incluso en aparatos de uso doméstico como: televisiones, teléfonos móviles, tabletas electrónicas, aparatos electrodomésticos, etc.

Las compañías de electricidad de todo tienen una incidencia de robos de energía eléctrica por parte de los consumidores que utilizan diversos mecanismos como toma clandestina y alteración del funcionamiento de los medidores. El porcentaje de perdidas debido a estos ilícitos se estima en algunos casos equivalentes al total de las perdidas debidas a otros factores, que llegan a sumar hasta 30% de la energía que se comercializa.

A pesar de lo grave del problema, apenas empieza a manifestarse una tendencia hacia la aplicación de los avances en dispositivos y técnicas de medición en la detección de robos de energía eléctrica y al control de su calidad, como la demuestran los sistemas de lectura automática de medidores.

La industria eléctrica no es la excepción a esta tendencia y en consecuencia, grupos de investigación, asociaciones profesionales, gobierno, universidades y empresas en particular, han desarrollado un concepto que abarca todo un conjunto de soluciones tecnológicas orientadas a modernizar el SEP (sistema eléctrico de potencia), combinando la infraestructura eléctrica construida para desarrollar una REI (Red Eléctrica Inteligente).

Se espera que la el sistema AMI aporte los siguientes beneficios:

1. *Confiabilidad en el suministro y calidad de energía*. Capaz de detectar anomalías en el sistema eléctrico y restablecerse de forma rápida y automática, afectando así al menor número de usuarios.
2. *Seguridad*. Todos los aparatos de importancia serán supervisados y analizados constantemente con la finalidad de detectar, prevenir y atender situaciones peligrosas que pudieran afectar la integridad del AMI.

3. *Uso eficiente de la energía.* Impulsa el uso de fuentes de energía renovables como solar y eólica por encima de métodos de generación convencional para disminuir emisiones de gases de efecto invernadero y otros contaminantes.
4. *Beneficios económicos.* Distribuye los gastos operativos por tratarse de un sistema automático. Además, proporciona información útil para desarrollar programas de mantenimiento preventivo y evitar daños mayores en activos.

2.2. - Definiciónes.

Las redes eléctricas inteligentes describen la próxima generación de sistemas eléctricos de potencia que se caracterizan por el incremento en el uso de comunicaciones e informática en la generación, distribución y consumo de energía eléctrica. Las tecnologías mencionadas mejoran principalmente la obsevabilidad y contabilidad del sistema eléctrico de potencia, pasando de ser una infraestructura estática, a una flexible y operada proactivamente. Propone también un área central de operaciones que analice y controle el resto de las áreas que integran el sistema eléctrico como se aprecia en la ilustración. (Institute E. P., 2009)

Todo este concepto engloba un conjunto de sistemas necesarios para actualizar el sistema eléctrico de potencia existente, utilizando sistemas de control, automatización, aplicaciones de procesamiento de información y redes de comunicación bidireccional. Todas estas tecnologías han sido utilizadas por décadas en la industria y empiezan a serlo los sistemas eléctricos.

La REI es una red eléctrica que inteligentemente integra acciones de todos los usuarios conectados a ella para distribuir la energía de forma eficiente, segura, económica y sustentable. (IEEE, 2012)

Ilustración 2. 1 Diagrama demostrativo de red eléctrica inteligente adaptado de (Institute E. P., 2009).

2.3. Modelo conceptual de la REI.

Este modelo conceptual proporciona una estructura de alto nivel para la REI que define 7 importantes dominios: generación, transmisión, distribución, consumidores, operaciones, mercados y proveedores de servicios. (Standards, 2010) que se aprecian en la ilustración 2-2

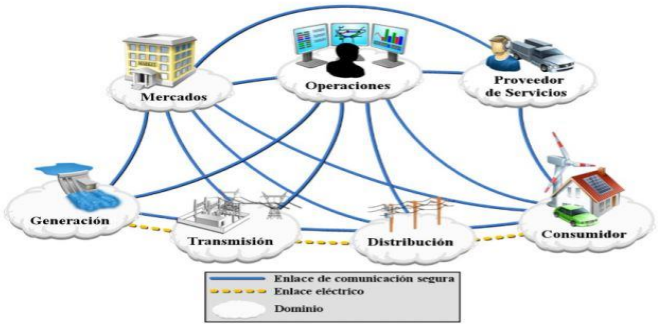


Ilustración 2. 2 Modelo conceptual de REI adaptado de (NIST, 2010

El modelo conceptual muestra también todas las comunicaciones y el flujo de la energía, conectando los dominios con diferentes tipos de enlaces para señalar la interacción entre cada uno de ellos.

Cada dominio está compuesto individualmente por elementos importantes de la red eléctrica inteligente que están también conectados entre sí por sistemas de comunicación bidireccional y de líneas eléctricas (NIST, 2010).

2.4. Lectura automática de medidores (AMR).

Automatic Meter Reading (AMR) cuyo significado en español es la lectura de medición remota, se ha convertido en una palabra de moda dentro del mundo de la medición (NIST, 2010).

AMR es un término aplicado a la variedad de tecnologías que permiten a las empresas se servicios públicos leer consumos de gas, agua y electricidad con mayor eficiencia. La mayoría de las empresas en Europa y estados unidos están migrando a sistemas AMR, en Asia e Hispanoamérica la mayoría de estas empresas ha empezado a realizar estudios para la

implementación de sistemas avanzados AMR de gran escala, mientras unos cuantos cuentan ya con dichos sistemas.

2.5. Infraestructura de medición avanzada.

El procesos de desarrollo del sistema de potencia eléctrico actual hacia la implementación de la REI será gradual y paulatino. No obstante, esta modernización ya empieza a vislumbrarse en algunos sectores de la industria eléctrica. Actualmente, los dominios de la generación y trasmisión ya son ampliamente supervisados e incluso controlados a distancia con el uso de los sistemas SCADA; sin embargo, la red de distribución no ha sido beneficiada con el mismo avance tecnológico, a pesar de que es el dominio donde se originan el 90 % de las fallas. (Farhangi, 2012).

Aunque ya se han realizado algunos intentos por desplegar equipos de medición e implementar un cierto grado de automatización en la red de distribución. Un ejemplo son los sistemas de medición conocidos como “Automated Meter Reading” (AMR).

La principal característica que distingue a la AMI de los sistemas de medición anteriores es precisamente la combinación de medidores digitales con la tecnología de comunicación bidireccional. Esto le brinda un gran potencial en cuanto a posibles aplicaciones de control en base a la información producida por los aparatos de campo. Una AMI involucra los siguientes sistemas.

- Medidores digitales inteligentes.
- Sistemas de comunicación bidireccional.
- Servidores que reciben y gestionan la información obtenida por los equipos de campo (Head End System).
- Software de control y análisis de datos para proporcionar información útil del sistema (MDM).

Todo el potencial de esta tecnología no será implementado inmediatamente, sino se irán agregando diferentes funciones, aprovechando gradualmente todos sus beneficios, antes de alcanzar la versión definitiva de la AMI. Algunas de las características de este proceso evolutivo se muestran a continuación en la ilustración 2.3.

Analizando la ilustración 2.3, se puede apreciar que gradualmente las características del AMI

Ilustración 2. 3 Evolución de los sistemas hacia REI adaptado de (Farhangi, 2012).

son muy similares al esquema planteado para una REI, donde los aparatos des sistema eléctrico son supervisados y controlados por una área central de operaciones. Es por ello que la AMI es considerada como la base de la REI.

2.5.1 Definición de AMI.

La infraestructura avanzada de medición (AMI) es el término usado para representar las tecnologías de sistemas de medición, que va hasta la administración automática remota de cargas de los consumidores por parte de las empresas de electricidad.

La base fundamental de las redes inteligentes está en la infraestructura avanzada de medidores (AMI) que está compuesta por medidores inteligentes o “Smart Meters” y un canal de comunicación bidireccional por medio de internet o redes similares entre los medidores y la Utility.

Los Medidores inteligentes proporcionan detalladamente y en tiempo real acerca de los consumos de los usuarios. Esta información permite a los proveedores, entre otras cosas, optimizar la generación de energía en función de la demanda, mientras que a los usuarios les permitirá reducir costes en sus facturas mediante un mejor conocimiento de sus consumos. Esto se muestra en la ilustración 2.4.

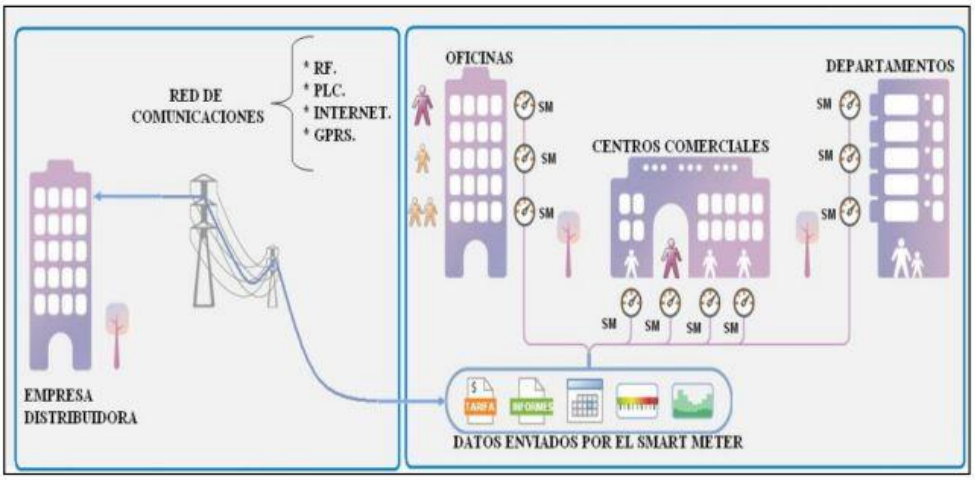


Ilustración 2. 4 Conexión de un sistema AMI, adaptado de (CFE, 2007).

2.5.2. Características de un sistema AMI.

Un sistema de medición inteligente tiene una gran variedad de características, entre las más destacadas se tienen:

- Mejora la calidad de la energía.
- Detección y ubicación de fallas de alimentadores eléctricos.
- Gestión de carga para carros eléctricos.
- Incorporación de electrodomésticos al concepto de red inteligente.
- Completamente bidireccional.
- Acceso a datos en tiempo real.
- Interoperabilidad de equipos M2M (Machine to Machine)
- Gestión y administración remota.

2.5.3. Beneficios con la implementación de sistemas AMI.

Un sistema de medición inteligente tiene beneficios tanto para el usuario como para la empresa proveedora de servicio.

Entre los beneficios para el consumidor una AMI brinda información en tiempo real acerca del consumo y el uso de la energía, así como los medios controlados y administrarlos de manera eficiente, reduciendo costos, mediante la activación de los controles de sus electrodomésticos inteligentes (refrigeradores, lavadoras, secadoras, aire acondicionado, iluminación, entre otros) de manera automática o manual además de la obtención de beneficios comunes, tales como: Eficiencia, confiabilidad, seguridad de la red y de servicio eléctrico, así como la protección del medio ambiente.

Y entre los beneficios para la empresa de electricidad un sistema AMI brinda mayor información e influencia sobre patrones de consumo y uso de la energía de sus clientes para una mejor predicción de la demanda permitiendo la tarifación en tiempo real. Mejorando así la eficiencia, confiabilidad, seguridad de la red y de servicio eléctrico, ahorros financieros por generación suspendida. Anticipación a fallas en la red eléctrica.

2.5.4. Sistemas principales de un sistema AMI.

Un sistema de medición avanzada consta de tres sistemas individuales:

- SON (Smart Objects Network) La red con tecnología Mesh, es una red en malla, utilizándola se pueden interconectar varios puntos de acceso (llamados nodos) y formar una malla de conexión que proporciona altas coberturas, es capaz de balancear carga de tráfico y es tolerante a fallos, de forma que si un nodo es retirado, la red puede auto configurarse para encontrar otras rutas de acceso.
- HES (Head End System) Es un sistema que es utilizado para configurar los dispositivos de la red de comunicaciones y medidores inteligentes, recolectar la información generada y monitoreo del estado de los dispositivos.

- MDM(Meter Data Management) Sistema que permite la validación, estimación y edición de los datos de medición recolectados por el Head End System y posterior envío hacia un sistema comercial o cualquier otro sistema (distribuidores de servicios, eléctrico, gas y agua)

2.6. Medidores inteligentes.

El medidor es una parte fundamental del sistema eléctrico, ya que vigila la trasferencia de energía en un punto específico del sistema (comúnmente en el punto de interconexión con el usuario). Anteriormente, el medidor electromecánico fue considerado, por muchos años como una obra maestra de la ingeniería, porque representaba de una forma económica, precisa, durable y sencilla de contabilizar el consumo de energía. Por tal motivo, su vigencia en el sector eléctrico fue casi de 100 años; sin embargo, carecía de funcionalidad. Era muy práctico para contabilizar el consumo total de energía, pero se volvía muy complicado para efectuar cualquier otro tipo de medición. Debido a esto, el medidor electromecánico ha sido gradualmente sustituido por el medidor digital, principalmente en usuarios industriales y recientemente a también a clientes residenciales (Institute E. P., 2010).

A diferencia del medidor electromecánico, el medidor digital muestra las formas de onda involucradas en la medición y las procesa mediante una serie de algoritmos lógicos y matemáticos. Estos medidores se implementan utilizando microcontroladores (MCU) o procesadores de señales digitales (DSP), donde se programan las funciones que desempeñará el dispositivo. El medidor digital es capaz de realizar cualquier tipo de medición dependiendo del algoritmo que tenga programado en si MCU o DSP, ya que prácticamente toma una fotografía de las señales que detecta (Institute E. P., 2009).

La ilustración 2.5 muestra un diagrama de bloques típico de un medidor digital, donde se aprecia como las señales de interés (voltaje y corriente) primero deben pasar a través de una etapa de acondicionamiento. Posteriormente las señales son digitalizadas por el convertidor analógico digital (ADC) y enviadas al microcontrolador como valores numéricos para realizar los cálculos pertinentes conforme al algoritmo previamente programado.

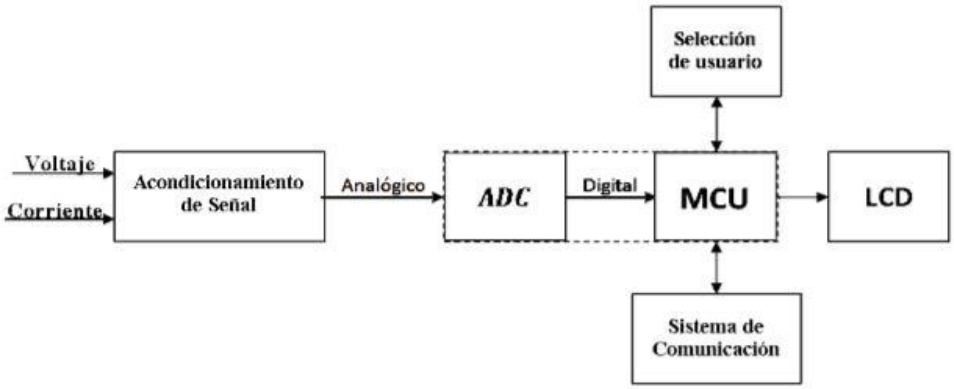


Ilustración 2- 1 Esquema básico de un medidor digital adaptado de (Institute E. P., 2010).

Las ventajas que ofrece un medidor digital con respecto al medidor electromecánico son:

1. Mayor precisión.
2. No presenta desgaste, ya que no tiene partes móviles (disco de inducción).
3. Capaz de detectar alteraciones o robo de energía.
4. Soporta diferentes tipos de mediciones ante cualquier factor de potencia.
5. Fácil calibración.

Adicionalmente, es posible agregar sistemas de comunicación al dispositivo, lo que permite ser supervisado, controlado e incluso reprogramado en forma remota. Se considera como medidor inteligente a aquellos medidores de comunicación bidireccional y vincular al usuario con la empresa suministradora del servicio, utilizando redes inalámbricas u otros medios de comunicación convencional (PG&E, 2012). Estos medidores poseen todas las virtudes del medidor digital, más la que proporciona un sistema de comunicación de estas características. La tabla 1 se menciona las principales ventajas de instalación de medidores inteligentes sobre medidores digitales sin comunicación.

Tabla 2. 1 Ventajas de la instalación de medidores inteligentes adaptado de (PG&E, 2012).

Áreas Beneficiadas	Ventajas

Consumidores	• Información para la gestión en el uso de energía. • Facturación precisa y oportuna. • Mejores opciones tarifarias. • Rápida restauración de fallas. • Información sobre la calidad de energía.
Operaciones de campo	• Menor costo en la lectura de medidores. • Reducción de toma de lecturas. • Elimina el uso de aparatos portátiles para la toma de lecturas. • Evita conexión y desconexión manual de servicios.
Facturación, contabilidad y protección de ingresos.	• Detección temprana de alteración de medidores o robo de energía. • Disminución de estimación en facturación. • Reducción de ajustes de cuenta por errores de facturación.
Transmisión y distribución	• Mejor manejo de carga en transformadores. • Mejor control de bancos de capacitores. • Información de eficiencia, confiabilidad, pérdidas y distribución de carga.
Continuacion de:	Tabla 2.1 Ventajas de la instalación de medidores inteligentes adaptado de (PG&E, 2012).
Mercados y pronósticos de carga	• Reducción de costos en recopilación de datos para estudios de carga.
Empresa suministradora del servicio	• Disminución de quejas. • Mejor perfil de riesgo. • Reducción en accidentes de operadores.
Beneficiarios externos	• Beneficios ambientales. • Favorece iniciativas de AMI.

2.7. - Sistemas de comunicación empleados en AMI.

Un componente crítico de la AMI es su sistema de comunicaciones (Gungor Vehbi, 2010). Se espera que los equipos de medición produzcan una gran cantidad de datos, por lo que es necesario definir los requisitos del sistema de comunicación, para seleccionar la mejor infraestructura de acuerdo a estas necesidades.

Existen diferentes tecnologías que soportan comunicación bidireccional y que pueden emplearse en una AMI, donde el objetivo principal es básicamente implementar dos flujos bidireccionales de información. El primer flujo viaja desde los sensores o dispositivos electrónicos hacia el medidor inteligente y el segundo flujo se presenta entre el medidor inteligente y los centros de información de la empresa eléctrica (Gungor Vehbi, 2010).

La Ilustración 8 ejemplifica los flujos de información que son propuestos en (Mera, 2011) y muestra los tres tipos de redes de comunicación (HAN, LAN y WAN) necesarias para establecer dicho enlace, así como algunos de los protocolos típicamente utilizados para cada tipo de red.

Ilustración 2. 5 Esquema de comunicación en sistemas AMI. Adaptado de (Mera, 2011).

En el ámbito de redes eléctricas inteligentes y sistemas de medición avanzada, el término “Home Área Network” (HAN) es empleado para describir una red local de corto alcance que comunica dispositivos digitales dentro de una residencia o edifico habitacional/comercial con medidor

inteligente. Estos dispositivos pueden ser termostatos, medidores de agua o gas, electrodomésticos, calentadores de agua, entre otros (Mera, 2011).

Una “Local Area Network” (LAN), o también conocida como “Neighborthoot Area Network” (NAN), es una red local pero con mayor alcance, cuyo objetivo es enlazar a las redes HAN formadas por los medidores inteligentes, es una misma área o localidad, como un concentrador de datos. Este último se encargará de reenviar toda la información recolectada a través de otro tipo de red (Mera, 2011).

Por otro lado, una “Wide Area Network” (WAN) abarca una amplia área geográfica como un estado o un país y comúnmente enlaza redes LAN más pequeñas. Dentro de una AMI, la WAN conecta los concentradores de datos con una estación central, utilizando tecnologías de comunicación como internet, GSM, GPRS, 4G o WiMax (Mera, 2011). Las tecnologías más populares para la implementación de redes tipo HAN, LAN y WAN en sistemas AMI se muestran en la siguiente tablas y posteriormente se explican las principales.

Tabla 2. 2 Tecnologías empleadas en redes HAN, LAN, y WAN en los sistemas AMI.

HAN	LAN	WAN
• **ZigBee** • **N-PLC**	• ZigBee • TWACS • GSM • Ethernet • Wi-Fi	• GPRS • 4G • WiMax • BPL • Internet Por Fibra Óptica

ZigBee. Es una especificación para un conjunto de protocolos de comunicación de alto nivel que utiliza radios digitales de baja potencia basados en el estándar IEEE 802.15. ZigBee se enfoca en aplicaciones de radiofrecuencia que requieren un bajo volumen de datos y bajo consumo de energía. La mayor virtud de este protocolo es su capacidad de formar redes de tipo de malla y sus propiedades de auto restauración. Utiliza también tecnología FHSS (Frequency Hoping Spread Spectrum) que proporciona inmunidad contra la interferencia y un buen desempeño a largo alcance. En los estados unidos, es el protocolo más utilizado para la implementación de redes HAN, principalmente debido a que importantes empresas trasnacionales, que ofrecen soluciones AMI, emplean este protocolo como estándar de sus productos (Laboratory, 2009).

N-PLC. Es un sistema de comunicación de bajo volumen de datos que opera a través de la instalación eléctrica de baja tensión de una casa o edificio. Es comúnmente empleada en aplicaciones de control dentro de las redes HAN. En Norteamérica, Japón y china, los dispositivos que emplean esta tecnológica operan a frecuencias cercanas a los 500 KHz, alcanzando una velocidad de transmisión de alrededor de 300 kbps. Por otro lado, en Europa, la operación de estos dispositivos está limitada a frecuencias menores a 148.5 KHz, por lo que su velocidad de transmisión reduce a 100 kbps (Zeev, 2012).

TWACS. Se trata de una tecnología patentada que utiliza la red de distribución eléctrica para establecer una comunicación bidireccional, de baja velocidad, entre diferentes puntos del sistema eléctrico. Estos sistemas normalmente comunican una subestación con el grupo de medidores para obtener información sobre el consumo de energía al usuario. Esta tecnología fue empleada en los primeros sistemas AMI y ha demostrado ser útil y funcional, sin embargo, requiere una significativa inversión al iniciar y sigue siendo un sistema de baja velocidad (TWACS, 2012).

Wi-Fi. Está basado en el estándar IEEE 802.11 y se trata de un protocolo efectivo para la creación de redes inalámbricas de banda ancha con velocidades desde 5 Mbps hasta los 56

Mbps, sin embargo, su alcance esta imitado alrededor de 100 metros, por lo que su despliegue regional sería muy costoso. Estas características hacen de este protocolo una opción atractiva desde el punto de vista técnico, aunque el costo de implementación pudiera resultar inaceptable (Institute E. P., 2009).

WiMax. Proporciona comunicaciones a larga distancia con un alcance de entre 10 y 30 millas y una velocidad de transmisión de 75 Mbps. Basada en el estándar 802.16, WiMax puede ser empleado como la columna vertebral de los sistemas de comunicación de los dominios de transmisión y distribución para aplicaciones de automatización de subestaciones o automatización de la red de distribución, así como proporcionar la infraestructura principal para los sistemas AMI (Institute E. P., 2009).

BPL. Es otra tecnología de comunicación que utiliza la infraestructura eléctrica para proporcionar comunicación de banda ancha a través de la red de media tensión. Efectiva en distancias alrededor de una milla, sin utilizar repetidores, puede alcanzar velocidades de transmisión entre 20 y 400 Mbps. Estándares para esta tecnología aún están actualmente en desarrollo, por lo que no han tenido buena aceptación en el mercado. (Institute E. P., 2009)

2.8. - Estándares involucrados con AMI.

Muchas aplicaciones, tecnológicas o soluciones orientadas a AMI han sido desarrolladas o se encuentran aún en desarrollo. Sin embargo, en un principio, muchos de estos desarrollos carecían de estándares ampliamente aceptados. Esta situación evitaba la integración de aplicaciones avanzadas, medidores y/o dispositivos inteligentes, integración de fuentes de energía renovables y la interoperabilidad entre estos sistemas. La adopción de estándares para todos estos sistemas es un prerrequisito crítico para que la AMI se convierta en una realidad. Los esfuerzos de estandarización pretenden alcanzar una interoperabilidad, seguridad de información, nuevos productos y sistemas, conjuntos de protocolos compactos e intercambio eficiente de la información (Sahin, 2011).

Diversas organizaciones de profesionistas como las NIST, IEC, IEEE, ANSI, ISO, ITU, etc. Han desarrollado una serie de estándares para diferentes áreas y marcan la pauta para el desarrollo de nuevos productos o servicios.

Algunos de los principales estándares relacionados con AMI aparecen en la siguiente tabla, así como una breve explicación de cada uno de ellos (Sahin, 2011).

Tabla 2. 3 Estándares Relacionados Con AMI.

Nombre del estándar	Descripción	Aplicación
ANSI C12.18	Protocolo de comunicación entre dispositivos que utilizan un puerto óptico ANSI serie 2.	AMI
ANSI C12.19	Define tablas de estructuras de datos para la transmisión de información en medidores inteligentes.	AMI
ANSI C12.20	Define criterios de construcción y precisión para medidores de estado sólido.	AMI
ANSI C12.22	Describe el protocolo para enviar tablas ANSI C12.19 a través de la red.	AMI
	Continuación de: Tabla 2- 3 Estándares Relacionados Con AMI.	
BACnet	Protocolo de comunicación para la automatización de edificios y aplicaciones como aire acondicionado, control de iluminación, alarmas etc.	Automatización de edificios
G3-PLC	Protocolo de comunicación a través de línea de potencia de baja y media tensión.	AMI
HomePlug Green PHY	Estándar para la creación de redes tipo HAN a través de línea de potencia.	HAN
IEC 60870-6	Protocolo de comunicación entre centros de control y aparatos desplegados en la red de transmisión.	Comunicación entre centros de control

IEC 61850	Estándar de comunicación entre aparatos de la red de transmisión, distribución y subestaciones.	Automatización de subestaciones
	Tabla 3- 2.3 Estándares Relacionados Con AMI. (Continuación)	
IEC 62351	Definiciones de seguridad para protocolos de comunicación.	Seguridad de comunicaciones
IEEE P1901	Comunicaciones de alta velocidad en línea de potencia.	HAN
IEEE P2030	Define estándares de interoperabilidad para la red eléctrica inteligente.	Aplicaciones para clientes
ITU-T G99.55	Establece la especificación de la capa física para transceptores de comunicación por línea de potencia OFDM de banda estrecha.	Automatización de red de distribución.
ITU-T G99.56	Establece la especificación de la capa de enlace de datos para el mismo tipo de dispositivos que ITU-T 99.55	Automatización de red de distribución.
M-Bus	Estándar europeo que establece especificaciones para la lectura remota de datos en medidores.	AMI
SAE J2293	Estándar para la transferencia de energía entre el sistema de potencia y autos eléctricos	Cargador para vehículos Eléctricos
SAE J2836	Define casos de uso para comunicación entre vehículos eléctricos y el sistema de potencia.	Vehículo Eléctrico
SAE J2847	Establece especificaciones de comunicación entre vehículos eléctricos y el sistema de potencia.	Vehículo Eléctrico
U-SNAP	Propone una interfaz modulas estándar que permite que diferentes productos sean compatibles con cualquier red a través de módulos de comunicación.	HAN
Z-Wave	Solución alternativa de ZibBee	HAN

2.8.1 - Estándar ANSI C12.18.

Este estándar detalla los criterios requeridos para la comunicación entre un dispositivo C12.18 y un cliente C12.18 a través de un puerto óptico. El cliente C12.18 puede ser un lector PDA,

una computadora portátil, una estación master o algún otro dispositivo electrónico de comunicaciones.

Cliente C12.18: Un aparato electrónico de comunicaciones que se conecta a través de un puerto óptico ANSI tipo 2 a un dispositivo C12.18 e implementa sus comunicaciones de acuerdo a las especificaciones de este estándar.

Dispositivo C12.18: Un aparato electrónico implementado de un puerto óptico ANSI tipo 2 para comunicaciones de acuerdo a las especificaciones de este estándar.

Comunicaciones punto a punto: Dentro del estándar C12.18 se define como comunicaciones punto a punto a la comunicación entre dos dispositivos a través de un simple interface óptico.

Tabla: Se define como una funcionalidad relacionada a elementos de datos, agrupados juntos en una simple estructura para ser transportados, esta estructura se especifica en la norma ANSI C12.19.

2.8.1.1. - Detalles del protocolo.

El protocolo describe en este estándar tres funcionalidades:

1. Establecimiento y modificación del canal de comunicaciones.
2. Transporte de información desde y hasta el dispositivo C12.18.
3. Cierre ordenado del canal de comunicaciones cuando la comunicación ha sido completada.

Tres capas del modelo OSI proveen las características de esta comunicación:

1. Capa física.
2. Capa de enlace de datos.
3. Capa de aplicación.

Con las condiciones por defecto establecidas en el estándar, el canal de comunicaciones es considerado siempre disponible una vez que la conexión física ha sido completada.

El servicio requerido de identificación es usado para obtener la versión del protocolo y la revisión e uso por el dispositivo C12.18.

Ciertos parámetros de comunicación pueden ser modificados a través del uso del servicio en la capa de aplicación. El servicio de negociaciones es opcional y si no es implementado en el dispositivo C12.18 o no es usado en la comunicación presente, los parámetros del canal de comunicaciones se quedan en los valores por defecto establecidos en este estándar.

Una vez que los parámetros de identificación y comunicación del dispositivo C12.18 han sido establecidos, la capa de aplicación provee los servicios de inicio de sesión (Logon), seguridad (Security), y cierre de sesión (Logoff). Los servicios de lectura y escritura para la trasferencia de datos, el servicio de terminación para desactivar el canal de comunicaciones y una estructura que provee información con respecto a los hechos y fallas de los servicios de petición.

En el orden de transmisión los parámetros en los diferentes servicios en todas las capas dentro de las definiciones del protocolo son transmitidos primero el byte más significativo. El orden de transmisión en los campos de datos y parámetros dentro de las tablas se define en la estructura de tablas, los bytes son transmitidos en tramas, cada trama tiene un byte de inicio.

2.8.2. - Estándar ANSI C12.21.

Este estándar detalla los criterios necesarios para las comunicaciones entre un dispositivo de medición de energía eléctrica y un host a través de un modem conectado a la red. El host puede ser una computadora portátil, una estación master, otro dispositivo de medición de energía eléctrica, o algún dispositivo de comunicaciones electrónicas. También proporciona información para la implementación del modelo OSI de 7 capas. El protocolo especificado fue diseñado para el transporte de datos en formato de tablas. Las definiciones de tablas se hacer referencia en la norma ANSI C12.19.

2.8.2.1 - Detalles del protocolo.

Siguiendo los lineamientos establecidos por el modelo OSI de siete capas, este protocolo ofrece tres funciones, estas son las siguientes:

1. La modificación del canal de comunicación.
2. El transporte de información desde y hacia el dispositivo de medición.
3. El cierre ordenado del canal de comunicaciones.

Las tres capas que se utilizan para proporcionar estas capacidades de comunicación.

1. Capa física.
2. Capa de enlace de datos.
3. Capa de aplicación.

En esta norma no se incluyen las definiciones para la creación del canal de comunicación. El canal de comunicación se considera disponible una vez que la conexión de la red se ha establecido con éxito y los módems se han sincronizado. El servicio de identificación PSEM es necesario establecer la versión del protocolo y la revisión de uso.

Ciertos parámetros de comunicación pueden ser modificados a través del uso de los servicios de negociaciones y de sincronización (Timing Setup) SPEM en la capa de aplicación. Estos servicios son opcionales y si no se aplican en el dispositivo de medición o no se utiliza durante la comunicación real, los parámetros de canal de comunicación se mantendrán en los valores predeterminados especificados por este estándar.

2.8.3. - Protocolo PSEM (Protocol Specification for Electricity Metering).

Este protocolo está definido tanto en el estándar C12.18 y C12.121, con variaciones para cada uno, se establece el lenguaje y las especificaciones para la comunicación en la capa de aplicación.

La capa de aplicación proporciona un conjunto mínimo de servicios y estructuras de datos necesarias que soportan los dispositivos finales para fines de recuperación de información, configuración y programación.

El protocolo está diseñado para transportar estructuras de tablas, el lenguaje PSEM ha sido diseñado para proporcionar una interfaz entre un equipo terminal y un dispositivo de medición en un medio de comunicación punto a punto. El lenguaje PSEM consta de 12 servicios. Cada servicio se compone de una petición y una respuesta. Estos servicios se muestran en la siguiente tabla.

Tabla 2. 4 Servicios definidos en PSEM.

ITEM	Servicio	Soporta ANSI
1	Identificación	C12.18, C12.21
2	Lectura	C12.18, C12.21
3	Escritura	C12.18, C12.21
4	Inicio de sesión	C12.18, C12.21
5	Seguridad	C12.18, C12.21
6	Cierre de sesión	C12.18, C12.21
Continuacion de:	Tabla 2- 4 Servicios definidos en PSEM.	
7	Negociación	C12.18, C12.21
8	Espera	C12.18, C12.21
9	Terminación	C12.18, C12.21
10	Configuración de tiempos	C12.21
11	Desconexión	C12.21

12	Autenticación	C12.21

2.9. - Sistema HES (Head End System).

La aplicación web hará uso de la estructura de la arquitectura de software llamada “Modelo 2”. Esta arquitectura de software se utiliza para realizar páginas JSP y es realmente el modelo vista controlador (MVC) enfocado para el diseño de aplicaciones web. Es por eso que los dos términos tanto “Modelo 2” y “MVC” se pueden utilizar indistintamente en el mundo web. La (SEO, 2010) arquitectura modelo 2 y sus derivados son la piedra angular para todas las aplicaciones web dado que da una estructura modular con la cual da fuerza al sitio web.

2.9.1. - Modelo Vista Controlador (MVC).

Model-View-Controller (Vitoria, 2007) (MVC) es un patrón de diseño, usado en aplicaciones web que necesitan tener la capacidad de mostrar múltiples vistas de los mismos datos (La base de datos). El MVC está definido por tres partes: la primera (el modelo) se encarga de los datos, la segunda (las vistas web) que tienen como función mostrar una parte o la totalidad de los datos y por último el controlador encargado de los eventos que efectúan a los dos anteriores.

- **Modelo:** El modelo define los datos que se utilizaran en la aplicación. Es la única parte del sistema que habla con la base de datos (por medio de Hibernate).
- **Vista:** La vista muestra los datos al usuario. La vista no hace ninguna transformación de los datos, solo presenta los datos. Usualmente en múltiples vistas.
- **Controlador:** El controlador es el programa que une a las vistas y el modelo. Toma la entrada del usuario y se da cuenta de lo que significa para el modelo que se actualice a sí mismo, y hace que el nuevo estado del modelo esté disponible para la vista.

Como se muestra en la Ilustración 2.9

- 1. El controlador recibe las peticiones del usuario.
- 2. El modelo se encarga de las consultas en base de datos.

- 3. El controlador toma los datos obtenidos y los envía a la vista, con lo que el usuario obtiene respuesta.

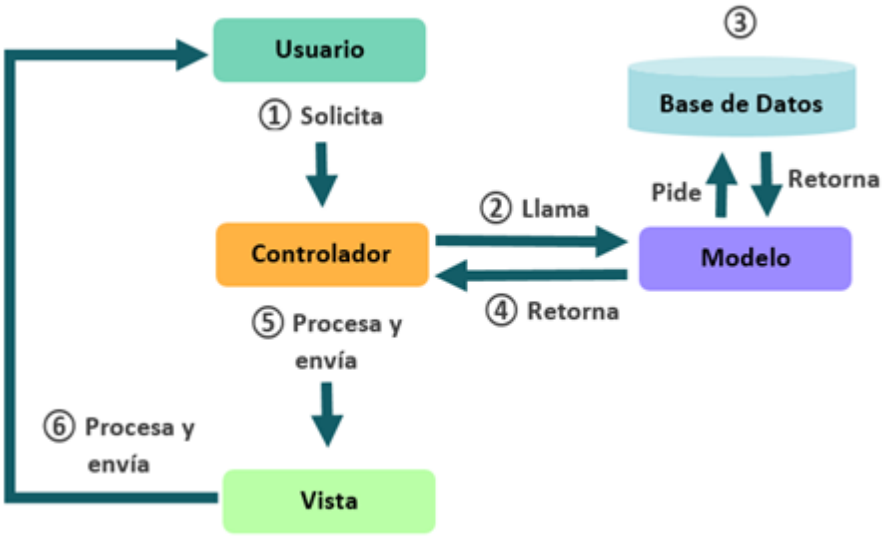


Ilustración 2. 6 Ciclo de vida del MVC. Adaptado de (Gómez, 2015).

2.9.2. - Bases de datos.

Las bases de datos son parte fundamental de los sistemas de cómputo modernos. Dado que permiten un manejo dinámico de la información, proporcionan seguridad, precisión y control; además de proveer elementos relevantes para la toma de decisiones operativas y tácticas relacionadas con el manejo de grandes volúmenes de información (Suárez, 2009).

Una base de datos es, un conjunto estructurado de datos guardados sobre soportes accesibles por la computadora, para satisfacer simultáneamente varios usuarios de manera selectiva y en un tiempo razonable (Suárez, 2009).

Las bases de datos tienen objetivos muy específicos (Integridad de los datos, confidencialidad, concurrencia de acceso, seguridad de funcionamiento) y todas esas funciones son controladas por un "Sistema Gestor de Base de Datos", SGBD, que es un software encargado específicamente de que la base de datos funcione correctamente.

Ilustración 2. 7 Sistemas gestores de bases de datos más utilizados.

Una base de datos puede ser definida como una colección estructurada y no redundante de datos relacionales que los asocian, almacenada sobre soportes accesibles por computadora y destinada a realizar numerosas aplicaciones.

Una base de datos está realizada para almacenar hechos del mundo real o simplemente de datos y es capaz de generar nuevos conocimientos.

Una base de datos se dice que es inteligente si tiene ciertas propiedades de inteligencia humana, es decir, que tenga una interfaz inteligente, donde el usuario pueda interactuar con la computadora de manera casi natural.

2.9.3. - Modelo relacional.

El modelo relacional fue propuesto E.F.Codd en los laboratorios de IBM en california, es el modelo más empleado, dado su versatilidad y potencia (Kendall, 2007).

Un modelo relacional es un modelo de datos basado en el concepto matemático de relación. Una relación es una parte del producto cartesiano de una lista de atributos, en donde cada atributo representa un valor de dominio.

Un dominio en la teoría de relaciones es un conjunto de valores posibles que puede tomar un atributo en nuestra base de datos, por ejemplo, la edad de un estudiante podría estar comprendida estrictamente en el intervalo (19, 26). Los dominios de una relación son necesariamente distintos. Si se designa por D1, D2, …, Dn. Un secuencia ordenada de "n" dominios, la relación seria una parte del producto cartesiano D1x, D2x … x Dn, los elementos de la relación de grado "n" son tuplas con notación (d1, d2, … , dn) en donde cada "di" toma valor en su dominio correspondiente (Kendall, 2007).

En el modelo relacional:

- Las bases de datos relacionales almacenan datos y resultados en las tablas.
- Las tablas están compuestas de filas y columnas.
- Cada tabla tiene una llave primaria, que es un identificador único de la tabla; la llave primaria puede estas compuesta por una o varias columnas.
- Para establecer relaciones entre las tablas es necesario incluir la llave primaria de una tabla A en una tabla B, en la tabla B la columna tomará el nombre de la llave foránea.

2.9.4. - Interfaz.

Para establecer la comunicación entre la base de datos y la página web, se configurara con Hibernate.

Hibernate utiliza dos tipos de configuración, la primera, la conexión con la base de datos, la segunda, ayuda a crear POJOS y archivos de mapeo (En la parte de implementación se explicará ampliamente).

2.9.5. - Controlador.

Ya que se tiene la comunicación con la base de datos y el modelo, la clase java llamada controlador se encarga de unir a las vistas y el modelo. El controlador ve todos los eventos

solicitados por el usuario, hace consultas con la ayuda de una clase y regresa la respuesta la cual será formada en el navegador por medio de JSP.

2.10. - Herramientas de software para la implementación de la aplicación.

Para el desarrollo de la aplicación web se utilizan las siguientes herramientas de software; las cuales están divididas según el Modelo Vista Controlador

Tabla 2. 5 Herramientas de software enmarcadas en el modelo vista controlador.

Modelo Vista Controlador	Herramienta de software.
Modelo	La base de datos la cual será administrada con Oracle Database XE 11.2 y la comunicación entre la base de datos y la aplicación será administrada mediante Hibernate formando el modelo.
Vista	Está formado con Java Server Faces con el cual utilizan la tecnología bootstrap.
Controlador	Para mostrar la información dentro de la página se utilizan clases POJO con el controlador de la tecnología de Java.

2.10.1. - Sistema Manejador de bases de datos (Oracle).

Las bases de datos relacionales almacenan y muestran resultados en tablas, para añadir, acceder, y procesar los datos almacenados en una base de datos, necesitamos un sistema manejador de base datos o gestor de base de datos (Piattini, 1999).

Oracle es un sistema de administración de base de datos (Database Management System) para base de datos relacionales. Oracle no es más que una aplicación que permite gestionar archivos

llamados de base de datos. Oracle como base de datos relacional utiliza múltiples tablas para almacenar y organizar la información.

Oracle Database 11g ofrece soluciones optimizadas para la gestión de base de datos que requieran escalabilidad, confiabilidad y alto desempeño empresaria (Piattini, 1999).

2.10.2. - Eclipse IDE para entorno de desarrollo integrado.

Eclipse es un entorno de desarrollo integrado de código abierto y multiplataforma. Mayoritariamente se utiliza para desarrollar aplicaciones de cliente, entorno de desarrollo integrado basadas en navegadores. Es una potente y completa plataforma de programación, desarrollo y compilación de elementos tan variados como sitios web, programas en C++ o aplicaciones en java. No es más que un entorno de desarrollo integrado (IDE) en el que encontraras todas las herramientas y funciones necesarias para desarrollo de aplicaciones, además de una interfaz lo que lo hace fácil de usar (Galli, 2005).

2.10.2.1. - Las características de Eclipse IDE.

- Dispone de un editor de texto con resaltado de sintaxis donde puedes Ver el contenido del fichero en el que estás trabajando.
- Contiene una lista de tareas y otros módulos similares.
- La compilación es en tiempo real.
- Tiene pruebas unitarias con JUnit.
- Integración con Ant, asistentes (wizards) para creación de proyectos, clases, tests, etc., y refactorización.

Si bien las funciones de Eclipse son más bien de carácter general, las características del programa se pueden ampliar y mejorar mediante el uso de plug-ins. Asimismo, a través de estos "plugins" libremente disponibles es posible añadir un sistema de control de versiones a través de Subversion y a la vez lograr una integración mediante Hibernate (Galli, 2005).

2.10.2.2. - Ventajas de su utilización.

- El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la Plataforma de Cliente rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.
- Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Python, permite a Eclipse trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos.
- La arquitectura plug-in permite escribir cualquier extensión deseada en el ambiente, como sería Gestión de la configuración. Se provee soporte para Java y CVS en el SDK de Eclipse. Y no tiene por qué ser usado únicamente para soportar otros Lenguajes de programación.
- La definición que da el proyecto Eclipse acerca de su Software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular".

En cuanto a la utilización de eclipse para la creación de aplicaciones clientes se puede decir que:

- Eclipse provee al programador con Frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de Software, Aplicaciones web, etc. Por ejemplo, GEF (Graphic Editing Framework - Framework para la edición gráfica) es un plug-in de Eclipse para el desarrollo de editores visuales que pueden ir desde procesadores de texto WYSIWYG (What You See Is What You Get) hasta editores de diagramas UML, interfaces gráficas para el usuario (GUI), etc. Dado que los editores realizados con GEF "viven" dentro de Eclipse, además de poder ser usados conjuntamente con otros plugins, hacen uso de su interfaz gráfica personalizable y profesional.
- El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código.

- El IDE también hace uso de un espacio de trabajo, en este caso un grupo de meta data en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente.

2.10.3. - Lenguaje de programación Java.

El lenguaje de programación Java está diseñado para satisfacer los desafíos de desarrollo de aplicaciones en un amplio entorno, desde aplicaciones de escritorio hasta lo que son entornos de red de amplia distribución. Teniendo como meta que estos consuman un mínimo de recursos del sistema, pueden ejecutarse en cualquier plataforma (hardware y software) y se puede ampliar de forma dinámica (Alegsa, 2009).

2.10.3.1. - Principales características de java.

Las principales características del lenguaje de programación en java son:

- **Lenguaje gratuito:** creado por SUN Microsystems absorbido por Oracle, que distribuye gratuitamente el producto base, denominado JDK (Java Development Toolkit) o actualmente J2SE (Java 2 Standard Edition). API distribuida con el J2SE muy amplia.
- **Simple:** El crecimiento masivo de internet nos lleva a una forma completamente nueva de ver el desarrollo y la distribución del software. La tecnología Java permite el desarrollo seguro, robusto y aplicaciones multi-plataformas de forma heterogénea y de redes distributivas. El sistema surgió para satisfacer estas necesidades, es sencillo, por lo que se puede programar fácilmente por la mayoría de los desarrolladores, de una manera familiar de modo que los desarrolladores actuales pueden aprender fácilmente el lenguaje de programación Java.
- **Orientado a objetos:** Java fue diseñado como un lenguaje orientado a objetos.18 Los objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos(o funciones) que manipulan esos datos. La tendencia del futuro, a la que Java se suma,

apunta hacia la programación orientada a objetos, especialmente en entornos cada vez más complejos y basados en red.

- **Distribuido:** Se dice que Java es distribuido ya que tiene un amplio API de rutinas para manejo fácil de los protocolos TCP/IP como HTTP y FTP. Con estas rutinas los programadores pueden acceder a los objetos mediante una URL con tal facilidad como si fuera un archivo local de una PC.
- **Dinámico:** El lenguaje Java y su sistema de ejecución en el tiempo real son dinámicos en la fase de enlazado. Las clases solo se enlazan a medida que son necesitadas. Se pueden enlazar nuevos módulos de código bajo demanda, procedente de fuentes muy variaras, incluso desde internet.
- **Arquitectura neutral y portable:** En el lenguaje Java, todo el código fuente es escrito en un archivo que tiene la extensión ".Java", los archivos se compilan y entonces se genera un archivo ".class".
- **Robusto:** Java está diseñado para crear un software de alta fiabilidad, ofrece un primer nivel de comprobación en el tiempo de compilación, seguido de un segundo nivel de comprobación en el tiempo de ejecución. Con esto podemos desarrollar software confiable. Sus características de memoria liberan a los programadores de una familia entera de errores (la aritmética de punteros), ya que se ha prescindido por completo los punteros y la recolección de basura elimina la necesidad de liberación explicita de memoria.
- **Multihilo:** Java soporta sincronización de múltiples hilos de ejecución a nivel de lenguaje, especialmente útiles en la creación de aplicaciones de red distribuidas. Así, mientras un hilo se encarga de la comunicación, otro puede interactuar con el usuario mientras otro presenta una animación en la pantalla y otro realiza cálculos.
- **Seguro:** La tecnología java está diseñada para funcionar en entornos distribuidos, lo que significa que la seguridad es de vital importancia. Java permite construir aplicaciones las cuales no pueden ser invadidas desde afuera. En el entorno de red, las aplicaciones escritas en lenguaje java son seguras, evitando intrusos.

2.11. - Servidor TomCat.

TomCat es un contenedor de Servlet que se utiliza en la referencia oficial de la implementación para Java Servlet y Java Server Pages (JSP). Las especificaciones Java Servlet y JavaServer Pages son desarrolladas por Sun Microsystems cuyas especificaciones vienen dadas por la JCP (Java Community Process). Apache TomCat es desarrollado en un entorno abierto bajo la licencia de Apache Software License (Foundation, 2018).

TomCat es un servidor web con soporte de servlets y JSPs. TomCat no es un servidor de aplicaciones, como JBoss o JOnAS. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de TomCat a menudo se presenta en combinación con el servidor web Apache.

TomCat puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de TomCat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y TomCat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad (Foundation, 2018).

2.12. - Java Server Pages (JSP).

Es un lenguaje para la creación de sitios web dinámicos, JSP acrónimo de Java Server Pages, está orientado a desarrollar páginas web e java. JSP es un lenguaje multiplataforma, creado para ejecutarse del lado del servidor. JSP fue desarrollado por Sun MicroSystems, Comparte ventajas similares a las de ASP.NET desarrollado para la creación de aplicaciones web potentes (Oracle, 2017).

La tecnología de JSP permite a los desarrolladores y a los diseñadores de web desarrollar rápidamente y mantener fácilmente páginas dinámicas, ricas en información como son las que soportan a sistemas de negociación. La tecnología de los JSP separa la interfaz del usuario de la parte lógica del contenido permitiendo a los diseñadores cambiar a su disposición las plantillas de la interfaz sin alterar el contenido dinámico subyacente (Oracle, 2017).

JSP también permite introducir código para la generación dinámica de HTML dentro de una página web. Esta surge por la necesidad de crear aplicaciones dinámicas para web de forma fácil, ya que la mejor parte del resultado de un programa CGI es estático. Se podría pensar entonces en JavaScript, pero este genera HTML dinámicamente en el cliente y no tiene acceso a los recursos del servidor, las principales características de JSP son:

- Conjunta el poder de Java en el servidor y la flexibilidad de HTML en el Browser.
- No solo se puede utilizar HTML, sino también XML o WML.
- Haces más fácil reusar componentes de JavaBeans los cuales realizan tareas más específicas.
- Su función es saber cómo procesar una solicitud para crear una propuesta.
- Soporta contenido dinámico que refleja las condiciones del mundo real.
- Es más rápido y fácil crear aplicaciones web.
- Capaz de instanciar cualquier clase java.

2.13. - Puente Ajax (Ajax Bridge).

El puente Ayax tiene como elementos un servidor residente y un cliente residente que coordinan la comunicación basada en Ajax entre el cliente navegador y el servidor en donde se encuentra la aplicación. El puente es responsable de entregar los cambios que se aumentan en la presentación desde la fase de formado al cliente del navegador, y re-ensamblar estos cambios en el navegador DOM para afectar los cambios de presentación. El puente también es responsable de detectar la interacción del usuario con la presentación y entregar al usuario eventos de vuelta a la aplicación para el procesamiento a través del ciclo de vida de JSP estándar. Un mecanismo llamado “Entrega parcial” es construir dentro del ICE faces generación de eventos automáticos a través del puente, entonces el desarrollador de la aplicación no es expuesto al mecanismo de evento de bajo nivel. El puente Ajax es estabilizado automáticamente a la primera página cargada de la aplicación y coordina las actualizaciones de la presentación y la transmisión de eventos de usuario por el tiempo de vida entero de la aplicación (Tools, 2017).

2.14. - Hibernate.

Hibernate es una herramienta ORM completa que ha conseguido en un tiempo record una excelente reputación en la comunidad de desarrollo posicionándose claramente como el producto OpenSource líder en este campo gracias a sus prestaciones, buena documentación y estabilidad. Es valorado por muchos incluso como solución superior a productos comerciales dentro de su enfoque, siendo una muestra clara de su reputación y soporte la reciente integración dentro del grupo JBoss que seguramente generará iniciativas muy interesantes para el uso de Hibernate dentro de este servidor de aplicaciones (Crespo, 2007).

Se empezó a desarrollar hace algo unos años por Gavin King siendo hoy Gavin y Christian Bauer los principales gestores de su desarrollo.

Hibernate parte de una filosofía de mapear objetos Java "normales", también conocidos en la comunidad como "POJOs" (Plain Old Java Objects), no contempla la posibilidad de automatizar directamente la persistencia de Entity Beans tipo BMP (es decir, generar automáticamente este tipo de objetos), aunque aún asi es posible combinar Hibernate con este tipo de beans utilizando los conocidos patrones para la delegación de persistencia en POJOs (Crespo, 2007).

Una característica de la filosofía de diseño de Hibernate ha de ser destacada especialmente, dada su gran importancia: puede utilizar objetos java definidos por el usuario tal cual, es decir, no utiliza técnicas como generación de código a partir de descriptores de modelos de datos de manipulación de bytecodes en el tiempo de complicación, ni obliga a implementar interfaces determinados, ni heredar una superclase. Utiliza en vez de ello el mecanismo de reflexión de Java.

Las razones que hacen que el uso de Hibernate sea muy importante son:

- **Simplicidad y flexibilidad:** necesita un único fichero de configuración en tiempo de ejecución y documento de mapeo para cada aplicación. este fichero puede ser el estándar de Java o un fichero XML. También se tiene la alternativa de realizar la configuración de forma programática. El uso de frameworks de persistencia, tales como EJBs hace que la aplicación depende del framework. Hibernate no crea esta dependencia adicional. Los

objetos persistentes en la aplicación no tienen que heredar de una clase de Hibernate u obedecer a una semántica especifica. Tampoco necesita un contenedor para funcionar.

- **Completo:** Ofrece todas las características de orientación a objetos, incluyendo la herencia, tipos de usuario y las colecciones. Además, también proporciona una capa de abstracción SQL llamada HQL. Las sentencias HQL son complicadas por el framework de Hibernate y cacheadas para su posible reutilización.
- **Prestaciones:** Una de las grandes confusiones que aparecen al utilizar este tipo de frameworks es creer que las prestaciones de la aplicación se ven muy mermadas. Este no es el caso de Hibernate. La clave en este tipo de situaciones es si se realizan el número mínimo de consultas a la base de datos. Muchos frameworks de persistencia actualizan los datos de los objetos incluso cuando no ha cambiado su estado. Hibernate solo lo hace si el estado de los objetos ha cambiado. El cacheado de los objetos juega un papel importante en la mejora de las presentaciones de la aplicación. Hibernate acepta distintos productos de cacheo, tanto de código libre como comercial.

2.14.1. - Tipos de objetos de Hibernate.

Hibernate distingue entre dos tipos de objetos:

- **Transient:** Sólo existen en memoria y no en un almacén de datos (recuérdese en este sentido también el modificador Transient de Java), en algunos casos, no serán almacenados jamás en la base de datos y en otros es un estado en el que se encuentran hasta ser almacenados en ella. Los objetos Transient han sido instanciados por el desarrollador sin haberlos almacenado mediante una sesión.
- **Persistent:** Se caracterizan por haber sido ya almacenados y por ser tanto objetos persistentes. Han sido creados y almacenados en una sesión o bien devueltos en una consulta realizada con la sesión.

2.14.2. - Sesión de Hibernate.

Para almacenar y recuperar estos objetos de la base de datos, el desarrollador debe mantener una conversación con el motor de Hibernate mediante un objeto especial, la sesión (clase Session).

Sirve para delimitar una o varias operaciones relacionadas dentro de un proceso de negocio, demarcar una transacción y aporta algunos servicios adicionales como una caché de objetos para evitar interacciones innecesarias contra la Base de Datos (Crespo, 2007).

Las sesiones son un concepto ligado a un proceso de negocio, por tanto es natural pensar que una sesión siempre va a pertenecer a un mismo thread de ejecución (el que pertenece a la ejecución de un método de negocio para un usuario o sistema externo concreto), aunque técnicamente se pueden compartir sesiones entre threads, esto no se debe hacer jamás por no ser una buena política de diseño y los consecuentes problemas que puede generar.

En un entorno multiusuario y por tanto multithread habrá por tanto múltiples sesiones simultáneas, cada una perteneciente a su correspondientes threads y con su contexto de objetos en caché, transacciones, etc.

Como tal no sorprende que las sesiones no son “thread-safe” y que la información vinculada a ella no sea visible para otras sesiones. Es también lógico que tenga que existir una “institución” superior para crear sesiones y realizar operaciones comunes a los diferentes threads como lo puede ser la gestión de una caché compartida entre threads o caché de segundo nivel (Crespo, 2007).

Este elemento es la clase SessionFactory y en ella podremos encontrar métodos como openSession() o evict (Class persistentClass).

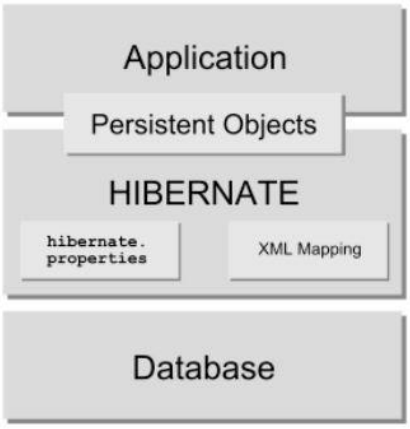


Ilustración 2. 8 Arquitectura básica de Hibernate adaptado de (Crespo, 2007).

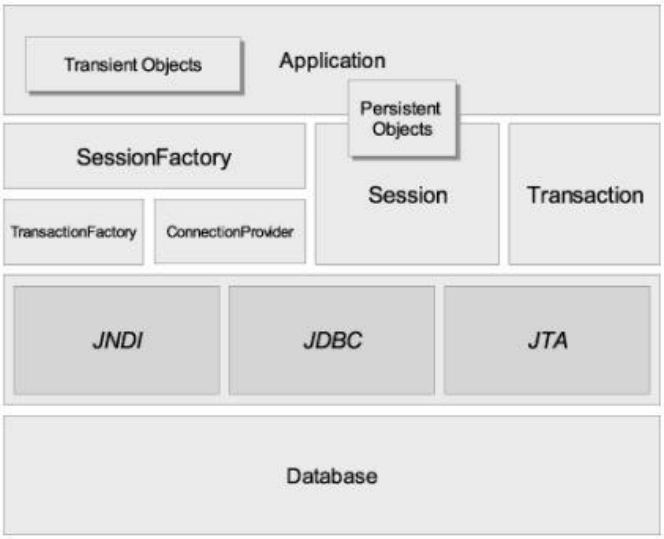


Ilustración 2. 9 Arquitectura de Hibernate completa adaptado de (Crespo, 2007).

2.14.3. - **Persistencia**.

La persistencia es un proceso por el que un objeto cualquiera se puede convertir en una secuencia de bytes con la que más tarde se podrá reconstruir el valor de sus variables. Esto permite guardar un objeto en un archivo o mandarlo por red (Cargnelutti, 2005).

Se dice que Hibernate es una máquina de persistencia ya que abre un canal de comunicación entre el modelo relacional y el modelo orientado a objetos.

Hibernate se encarga de la configuración de la conexión a la base de datos, además de crear las clases de persistencia (POJOS) y los archivos de mapeo (Cargnelutti, 2005).

2.14.4. - Lenguaje de consulta de Hibernate (HQL).

Las consultas a la base de datos, no serán hechas directamente en la base de datos, debido a que se mostraran en un ambiente de objetos, después de hacer un proceso (el proceso se verá paso a paso en el diseño de la interfaz) para que la base de datos que está en el mundo relacional puede ser trabajada en el mundo de los objetos. Este proceso será realizado mediante Hibernate. Al pasar por el proceso veremos a la base de datos como un objeto, entonces la tecnología Hibernate tiene su propio generador de consultas que trabaja con objetos.

Hibernate usa un lenguaje de consulta de gran alcance (HQL) que es similar en apariencia a SQL. En comparación con este, sin embargo, HQL es completamente orientado a objetos y comprende nociones de la programación orientada a objetos (Herencia y Polimorfismo).

2.15. - Apache Maven.

La construcción de aplicaciones Java EE es un proceso complejo en el cual deben realizarse tareas de construcción, empaquetado y despliegue en múltiples entornos. Estas aplicaciones suelen depender frecuentemente de un gran número librerías y frameworks de terceros. Apache Maven, es un framework de gestión de proyectos de software, que proporciona un modelo estándar de gestión y descripción de proyectos. Maven da soluciones a tareas que abarcan desde la compilación hasta la distribución, despliegue y documentación de los proyectos. Se podría describir como “un sistema de estándares, un repositorio, y un software utilizado para manejar y describir proyectos. Define un ciclo de vida estándar para la construcción, prueba y despliegue de componentes del proyecto. Proporciona un marco que permitirá la reutilización Fácil de la lógica común de la estructura para todos los proyectos que siguen los estándares maven”. El

principal objetivo de maven es que un desarrollador pueda adaptarse al método de trabajo de un proyecto en el menor tiempo posible, disminuyendo su curva de aprendizaje.

Para lograr este objetivo primordial de Maven se encarga de diversas materias:

- Facilita el proceso de construcción.
- Proporciona un sistema de construcción uniforme. Maven se basa en la definición de metadatos del proyecto, en el Project Object Model (POM), almacenados en un fichero denominado pom.xml, unida a la utilización de una serie de plugins comunes a los proyectos que utilizan Maven. Todos los proyectos Maven funcionan de la misma forma, por lo que el esfuerzo de aprendizaje sólo se hace una vez.
- Proporciona información útil sobre el proyecto. Partiendo de la información almacenada en el POM (Project Object Model) Maven puede gestionar automáticamente información del proyecto de gran utilidad como por ejemplo: Listas de cambios desde el control de versiones, dependencias transitivas, informe de la ejecución de pruebas unitarias.
- Ayuda a utilizar las “mejores prácticas” de desarrollo (Best Practices). Maven obliga a aceptar una serie de convenciones y costumbres que aportan claros beneficios. Por ejemplo, marca una estructura estándar para el código fuente, la documentación, existe un único sitio donde describir el proyecto, obliga a tener el código fuente de las pruebas unitarias de forma separada pero relacionada, etc.
- Permite introducir nuevos servicios de forma sencilla: plugins para la creación automática de un portal Web del proyecto fácilmente configurable, para el cálculo de métricas de calidad del código fuente, para su despliegue en los entornos de desarrollo, preproducción y producción, etc.

2.16. - JPA (Java Persistence API)

Java Persistence API (JPA) proporciona un modelo de persistencia basado en POJO´S para modelar base de datos relacionales en Java. El Java Persistence API fue desarrollado por EJB 3.0 como parte de JSR, aunque su uso no se limita a los componentes de software EJB. También

puede utilizarse directamente en aplicaciones web y aplicaciones clientes; incluso fuera de la plataforma de Java EE, por ejemplo en aplicaciones Java SE (Caules, 2009).

En su definición, se han combinado ideas y conceptos de los principales frameworks de persistencia como Hibernate, Toplink y JDO, y de las versiones anteriores de EJB. Todos estos cuentan actualmente con una implementación JPA.

El mapeo objeto/relacional, es decir, la relación entre entidades Java y tablas de la base de datos, se realiza mediante anotaciones en las propias clases de entidad, por lo que no se requieren ficheros descriptores XML. También pueden definirse transacciones como anotaciones JPA (Caules, 2009).

Java Persistence API consta de tres áreas:

- El java persistence API.
- El lenguaje query.
- El mapeo de metadatos objeto/relacional.

2.17. - Spring Security.

Spring Security es un framework de apoyo al marco de trabajo Spring, que dota al mismo de una serie servicios de seguridad aplicables para sistemas basados en la arquitectura basados en J2EE, enfocado particularmente sobre proyectos construidos usando SpringFramework. De esta dependencia, se minimiza la curva de aprendizaje si ya es conocido Spring (Software, 2018).

Los procesos de seguridad están destinados principalmente, a comprobar la identidad del usuario mediante la autenticación y los permisos asociados al mismo mediante la autorización. La autorización es dependiente de la autenticación ya que se produce posteriormente a su proceso (Software, 2018).

Por regla general muchos de estos modelos de autenticación son proporcionados por terceros o son desarrollados por estándares importantes como el IETF adicionalmente, Spring Security

proporciona su propio conjunto de características de autenticación. Específicamente, Spring Security actualmente soporta integración de autenticación con todas las siguientes tecnologías:

- HTTP BASIC authentication headers (an IEFT RFC-based standard).
- HTTP Digest authentication headers (an IEFT RFC-based standard).
- HTTP X.509 client certificate exchange (an IEFT RFC-based standard).
- LDAP (un enfoque muy común para necesidades de autenticación multiplataforma, específicamente en entornos extensos).
- Form-based authentication (necesario para interfaces de usuario simples).
- OpenID authentication.
- Computer Associates Siteminder.
- JA-SIG Central Authentication Service.
- Automatic "remember-me" authentication.
- Anonymous authentication.
- Run-as authentication.
- Java Authentication and Authorization Service (JAAS)
- Container integration with JBoss, Jetty, Resin and TomCat (tambien podemos usar autenticación gestionada por el contenedor)
- Java Open Source Single Sign On (JOSSO)*
- OpenNMS Network Management Platform*
- AppFuse*
- AdroMDA*
- Mule ESB*

Spring Security tiene un alto grado de aceptación entre la comunidad de desarrolladores por su gran flexibilidad sobre los modelos de autenticación. De esta manera, permite una rápida integración de las soluciones que presentan ante los requerimientos de los clientes potenciales, sin implicar una migración de los sistemas a un determinado entorno. Además es una plataforma abierta y en constante evolución, lo que permite ir añadiendo nuevos mecanismos de autenticación, o directamente programar los propios de manera poco compleja (Software, 2018).

En ocasiones un proceso simple de autenticación no es eficiente. Hay que controlar como se relaciona el usuario con la aplicación. Puede resultar interesante como llegan las solicitudes, si son automatizadas o por medio de una persona, asegurar que se realizan mediante el protocolo Http. Para completar estos objetivos, Spring Security proporciona "canales de seguridad" automáticos integrándose con JCaptcha para detección de usuarios humanos.

Spring Security también facilita el proceso de la autorización. Para ello, aporta tres características esenciales: Autorización en base a solicitudes web, autorización en base a llamadas, autorización en base al acceso a instancias (Software, 2018).

2.18. Metodología de desarrollo.

2.18.1. - ¿Qué es Scrum?.

Scrum es una metodología ágil para el desarrollo de software o la gestión de proyectos. Antes de la definición de Scrum, es imprescindible entender el concepto ágil (Scrum, 2017). El desarrollo de software ágil se define como:

Generalmente, el desarrollo de software es una actividad caótica, a menudo se caracteriza por la frase “código y arreglar”. El problema del desarrollo de software reside en que el código se escribe muchas veces si seguir un plan subyacente, y el propio diseño del sistema se improvisa a partir de muchas decisiones a corto plazo. De hecho, esto funciona bastante bien cuando el sistema es pequeño, pero conforme el sistema va creciendo, se va haciendo más difícil añadir nuevas funcionalidades. Además, también se hacen predominantes los bugs y aumenta la dificultad de arreglarlos. Para evitarlos, es necesario una larga fase de pruebas una vez que se han definido todas las funciones del sistema. Esta fase causa estragos en los cronogramas, ya que la realización de pruebas y la depuración es imposible planificar (Scrum, 2017).

El movimiento original de cambiar esto, introdujo la noción de metodología. Estas metodologías imponen un proceso disciplinado sobre el desarrollo de software con el objetivo de elaborar un software más predecible y eficiente. Esto se consigue desarrollando un proceso detallado con

especial énfasis en la planificación, inspirada en disciplinas de la ingeniería también llamadas metodologías de la ingeniería.

Las metodologías ágiles surgieron como reacción a estas metodologías. La atracción que sufre mucha gente por estas metodologías viene provocada por la burocracia que conlleva metodologías de la ingeniería. Estos métodos nuevos intentan ofrecer un compromiso eficiente entre procesos, proporcionando los procesos únicamente para lograr una recompensa razonable. El resultado de todo esto son los significativos cambios que sufren los métodos ágiles frente a los métodos de la ingeniería. La diferencia más determinante es que estas metodologías se basan en la poca documentación, normalmente resaltando la poca cantidad de documentos para una tarea. Se podría decir que están enfocadas al código (Scrum, 2017).

Las metodologías ágiles son flexibles más que predictivas. Los métodos de la ingeniería tienden a planificar detalladamente una gran parte del proceso del software durante un largo periodo de tiempo, lo que funciona hasta que algo cambia. En cambio las metodologías agiles se acogen al cambio continuo.

Por otro lado, las metodologías ágiles están más enfocadas al ser humano que a los procesos. El objetivo de las metodologías de la ingeniería es definir un proceso que funcionara bien sin importar quien lo vaya a usar. En cambio, las metodologías ágiles reivindican que el no usar procesos supondrá el desarrollo de las habilidades del equipo, de formar que, el papel de un proceso es dar soporte al equipo desarrollador en su trabajo (Scrum, 2017).

Actualmente hay muchas metodologías agiles en uso, pero ha sido necesario el uso de esta metodología para el desarrollo del presente proyecto. Por lo tanto es imprescindible hablar de Scrum antes de empezar el desarrollo.

A continuación se presenta una breve descripción de la metodología Scrum:

La metodología Scrum para el desarrollo ágil de software representa un punto de partida de la gestión en cascada. De hecho, Scrum y otro tipo de procesos agiles se inspiraron en sus limitaciones. La metodología Scrum enfatiza en la comunicación y colaboración, el funcionamiento del software, y la flexibilidad de la que dispone para adaptarse a las emergentes realidades de las empresas, todos los atributos de los que carecía el modelo de cascada.

2.18.2. - ¿Qué caracteriza a Scrum?.

De todas las metodologías agiles, Scrum es única porque introduce la idea del control empírico de los procesos. Esto significa que Scrum utiliza el proceso real de un proyecto para planificar y concertar los lanzamientos. En Scrum, los proyectos se dividen en ritmos de trabajo breves, conocidos como sprints. Normalmente, tiene una, dos o tres semanas de duración. Al final de cada sprint, el cliente y los miembros del equipo se reúnen para evaluar el proceso del proyecto y planear los siguientes pasos a seguir. Esto permite que la dirección del proyecto se ajuste o se oriente una vez finalizado el trabajo, sin especulaciones ni predicciones (Scrum, 2017).

Filosóficamente, este énfasis continuo de evaluar las tareas finalizadas es el principal causante del éxito que tiene esta metodología entre los directores y desarrolladores. Pero lo que verdaderamente permite funcionar a la metodología Scrum es un conjunto de papeles, responsabilidades y reuniones.

2.18.3. - Los papeles de Scrum.

Scrum tienen papeles fundamentales: Product Owner (Propietario del producto), Scrum Master (Especialista en Scrum) y el Team Member (Grupo de desarrolladores).

- **Product Owner:** En Scrum, el product Owner se encarga de comunicar la visión del producto al equipo de desarrollo. Él/Ella tambien deben presentar el interés del cliente por medio de los requisitos y la priorización. Como el product Owner es el que más autoridad tiene de los tres papeles de Scrum, tambien es el que mayor responsabilidad recibe. En otras palabras, el product Owner es el individuo que tiene que afrontar las consecuencias cuando el proyecto va mal.

La autoridad y responsabilidad del product Owner le hace difícil conseguir el balance correcto de implicación. Como Scrum evalúa la auto-organización entre equipos, el product Owner debe luchar por no supervisar hasta el último detalle. Al mismo tiempo, el product Owner tiene que estar en la disposición de las preguntas del equipo.

- **Scrum Master:** El Scrum master actúa como enlace entre el product Owner y el equipo. El Scrum master no dirige el equipo. Él/Ella se encarga de evitar cualquier barrera que impida al equipo lograr sus objetivos de sprint. En resumen, este papel hace que el equipo sea creativo y productivo, a la vez que permite que los logros del equipo sean visibles ante el product Owner sobre como maximizar el ROI (Return Of Investment) para el equipo.
- **Team Member:** En la metodología Scrum, el equipo es responsable de terminar el trabajo. Idealmente, los equipos están formados por 7 miembros multifuncionales. Para proyectos de software, el equipo habitual sería una mezcla de ingenieros de software, arquitectos, programadores, analistas, testers y diseñadores de ULs. En cada sprint, el equipo es responsable de determinar cómo va a lograr acabar el trabajo. Esto garantiza al equipo un grado de autonomía, pero, al igual que pasa con la situación del product Owner, esta libertad viene acompañada por la responsabilidad de cumplir los objetivos del sprint.

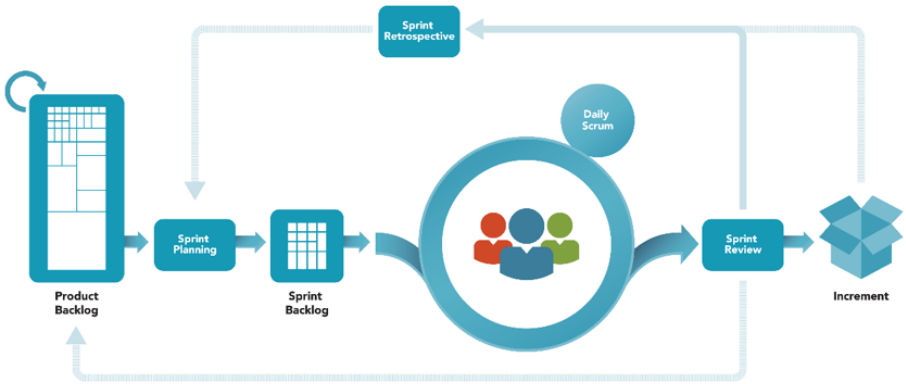


Ilustración 2. 10 Flujo habitual de Scrum adaptado de (Scrum, 2017).

2.18.4. - Análisis del sistema.

En este apartado, la meta a conseguir es identificar los objetivos que han de ser alcanzados para el desarrollo del sistema final. Para lograr esto se tomará como referencia las historias de usuario

que representan las necesidades que deben cubrir las funcionalidades de la aplicación, y de este modo satisfacer las exigencias del cliente (ScrumOrg, 2018).

Para la realización del proceso de extracción de información relativa a las necesidades a cubrir se llevara a cabo entre los miembros del esquipo de desarrollo y el propio cliente. Este proceso, gracias a la metodología Scrum, no se limitará a la fase inicial del proyecto, como ocurriría en un desarrollo clásico, si no que se tratara de un proceso iterativo en constante evolución para poder amoldarse a los requerimientos del cliente de la forma más eficiente.

Las historias del usuario que se extraigan de las reuniones con el cliente, describirán una funcionalidad que debe incorporar un sistema de software y cuya implementación aporta valor al cliente (ScrumOrg, 2018).

La estructura de una historia de usuario está formada por:

- Nombre breve y descriptivo.
- Descripción de la funcionalidad en forma de dialogo o monologo del usuario describiendo la funcionalidad que desea realizar.
- Criterio de validación y verificación que determinara para considerar terminado y aceptable por el cliente el desarrollo de la funcionalidad descrita.
- Las historias de usuario se descomponen en tareas. Estas tareas son unidades de trabajo que tienen asignado un esfuerzo estimado. Por lo que es en las tareas en los que se basa la estimación de esfuerzos generales del proyecto.

2.18.5. - Definición del sistema.

Para la definición del sistema, es decir, para la extracción de la información que ayude a modelar la aplicación según las necesidades del cliente, se realiza una reunión entre todos los miembros interesados en el proyecto. En dicha reunión el cliente expone su idea del sistema, que necesidades debe cubrir y que funciones quiere que le aporte. También los miembros del equipo técnico aportan su visión para enriquecer la definición final del sistema a desarrollar. Finalmente con toda la información extraída de la reunión el Product Owner, el actor que representa la voz

del cliente, escribe las historias de usuario donde se representa formalmente la definición del sistema, además se extraen las tareas que componen las historias de usuario, se les asigna un coste (un esfuerzo estimado) y las prioriza. Quedando formado el Product Backlog que será la base del proyecto a desarrollar (ScrumOrg, 2018).

Es importante aclarar que debido a la naturaleza ágil del proyecto, la definición inicial del sistema puede verse alterada en el transcurso del desarrollo del mismo. Ya que como más adelante se verá, el desarrollo se divide en diferentes etapas o Sprints, entre los cuales se repetirá la reunión entre los miembros del proyecto, pudiéndose incluir, modificar o eliminar historias de usuario y/o tareas. Por este motivo el análisis expuesto en este capítulo únicamente se trata del análisis inicial, y en próximos capítulos se describirán los análisis correspondientes a la reunión de cada Sprint (ScrumOrg, 2018).

2.18.6. - Historias de usuario.

Las historias de usuario se definirán utilizando el siguiente modelo:

Tabla 2. 6 Modelo para la historia de usuario adaptado de (ScrumOrg, 2018).

Historia de usuario	Historia de usuario
Id	
Nombre	
Prioridad	
Riesgo	
Descripción	
Validación	

Donde cada campo tiene el siguiente significado:

- **Id:** Se trata del identificador único asignado a este elemento del proyecto.

- **Nombre:** Es nombre corto utilizado para describir muy brevemente la historia de usuario.
- **Prioridad:** Es la preferencia de cara al desarrollo de la historia de usuario respecto a los demás, Valores: Alta, Media y baja.
- **Riesgo:** Se trata de la importancia de la historia de usuario en relación al conjunto del proyecto. Cuantificando de este modo el daño provocado en caso fallido. Valores: Alto, Medio y bajo.
- **Descripción:** Breve explicación de las intenciones de la historia de usuario, debe dejar la idea clara de la propia historia.
- **Validación:** son las condiciones que deben cumplirse una vez la historia está completamente desarrollada para que se dé por finalizada.

2.18.7. – Tareas.

Las tareas que componen las historias de usuario se detallan a continuación siguiendo el siguiente modelo:

Tabla 2. 7 Ejemplo de tabla de tarea adaptado de (ScrumOrg, 2018).

Tarea	
Historia de usuario	
Estado	
Descripción	

- **Tarea:** Es el identificador único para elemento del proyecto. El formato que sigue en Txx, donde las “xx” serán el número entero del 01 al 99.

- **Historia De Usuario:** Toda tarea forma parte de una historia de usuario que se indica en este campo.
- **Estado:** Se trata de la fase en la que se encuentra el desarrollo de la tarea. Con valores; No iniciada, En proceso, completada.
- **Descripción:** Una breve explicación de la finalidad de la tarea.

2.18.8. - Sprints.

En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones de un mes natural hasta de dos semanas). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto que sea potencialmente entregable, de manera que cuando un cliente (Product Owner) lo solicite solo sea necesario un esfuerzo mínimo para que el producto esté disponible para ser utilizado. Para ello, durante la iteración el equipo colabora estrechamente y se llevan a cabo las siguientes dinámicas (ScrumOrg, 2018).

- Cada día el equipo realiza una reunión de sincronización, donde cada miembro inspecciona el trabajo de los otros para poder hacer las adaptaciones necesarias, comunica cuales son los impedimentos con que se encuentran, actualiza el estado de la lista de tareas de la iteración (Sprint Blacklog) y los gráficos de trabajo pendiente.
- El facilitador (Scrum Master) se encarga de que el equipo pueda cumplir con su compromiso y de que no merme su productividad, elimina los obstáculos que no puede resolver por sí mismo y protege al equipo de interrupciones externas que puedan afectar su compromiso o su productividad.

También existen una serie de restricciones que deben ser consideración:

No se puede cambiar los objetivos/requisitos de la iteración en curso:

- En la reunión de planificación de la iteración el equipo adquirió el compromiso de completar en la iteración unos requisitos concretos, baso su plan y organización en

ellos. Cambiar los objetivos/requisitos de la iteración dificulta la concentración del equipo, baja su moral y su compromiso.

- El hecho de o poder cambiar los objetivos/requisitos de la iteración una vez iniciada facilita que el cliente cumpla con su responsabilidad de conocer que es lo más prioritario a desarrollar, antes de iniciar la iteración.
- Notar que Scrum minimiza esta necesidad ya que, por un lado, los objetivos/requisitos que se están desarrollando eran los más prioritarios justo antes de iniciar la iteración y, por otro lado, las iteraciones en Scrum son suficientemente cortas (2 o 4 semanas) como para que la probabilidad de cambios una vez iniciada la iteración sea mínima.

Terminación anormal de la iteración:

- Solo en situaciones muy excepcionales el cliente o el equipo pueden solicitar una terminación anormal de la iteración. Esto puede suceder si, por ejemplo, el contexto del proyecto ha cambiado enormemente y no es posible esperar al final de la iteración para aplicar los cambios, o si el equipo encuentra que es imposible cumplir con el compromiso adquirido. En ese caso, se dará por finalizada la iteración y se dará inicio a otra mediante una reunión de planificación de la iteración.

Capítulo 3. Desarrollo de la metodología ágil Scrum.

En el presente capitulo se muestra la implementación de la metodología para la creación de una base de datos relacional para un sistema Head End System que se encargue de la información que es enviada desde el sistema comercial de CFE hacia la red de medidores, así como la que es generada por los medidores dentro de la red inteligente en respuesta de las operaciones ejecutadas, así como la implementación de Spring security sobre la interfaz gráfica de usuario, que se encargue de la seguridad de información a nivel base de datos y URL´s únicas en el sistema, para el desarrollo de la metodología propuesta cada sprint fue dividivo en pequeñas tareas y posteriormente el capitulo 4 de esta tesis se muestra el resultado de las tareas mas sobresalientes de cada sprint.

3.1. - Requisitos del Product Owner para un sistema HES.

En la primera reunión con el Product Owner se definieron las características mínimas con las que debe cumplir el sistema AMI, las cuales se describen a continuación.

El centro de gestión AMI debe cumplir con lo siguiente:

- El protocolo TCP/IP deben cumplir con las versiones 4 (IPv4) y/o 6 (IPv6).
- Las características de seguridad cibernética del sistema por el medio se software acorde con lo especificado por CFE para asegurar que solamente los usuarios autorizados pueden acceder al sistema, a sus datos y funciones.

El sistema informático HES debe cumplir con lo siguiente:

- Debe soportar acceso de al menos 30 operadores simultáneamente, así como asegurar su funcionamiento.
- Debe ser capaz de operar la totalidad de medidores en cualquier gabinete que conforme la red mesh.
- Debe ser capaz de acceder remotamente, a través de claves de acceso con encriptación AES128, en cualquier momento, a todos los medidores en cualquier área de ubicación.
- Debe ser capaz de recolectar de manera automática programada y a petición del operador, desde un medidor hasta un grupo de medidores.

- Debe ser capaz de enviar de manera programada y/o a petición del operador, sincronía de tiempo, instrucciones de corte y conexión a un medidor, en cualquier área geográfica de la red.
- El sistema de gestión HES sebe ser capaz de recibir la alarma de un medidor o grupo de medidores.
- Ser Capaz de almacenar los datos de lecturas, alarmas y eventos, de por lo menos 2 años en línea.
- La base de datos debe soportar lenguajes de consulta estructurados.
- Ante una interrupción de suministro de energía eléctrica el sistema se debe reiniciar en forma automática y funcionar en un tiempo máximo de 30 minutos.
- Tener capacidad de realizar automáticamente el cambio de horario estacional y manejar al menos 4 zonas horarias.
- Debe soportar tareas programadas en rutas de trabajo que requieran lectura de registro.
- Para el caso de los medidores en gabinete debe ser capaz de asociar uno, dos o tres medidores a un cliente.
- Expedir reportes de lecturas de registros, eventos, alarmas, perfiles de carga, estatus de la red.
- Capaz de proporcionar informes de lecturas de energías y demanda de un medidor, un grupo de medidores, con base a un rango de fechas.
- Capaz de proporcionar listado de medidores asociados a una tarea programada incluyendo los porcentajes de lecturas adquiridas y no adquiridas al día actual.
- Capaz de mostrar la cantidad de medidores asociados a una tara programada incluyendo los porcentajes de lecturas adquiridas al día actual.
- Estadística diaria de los porcentajes de lecturas adquiridas con base a un rango de fechas.

Listado de medidores que presentaron alguna clase de evento, el cual debe ser seleccionable por el usuario y en base a un rango de fechas también especificado por el usuario. En detalle de la información se deben incluir las cedulas a los cuales se encuentran asociados los medidores así como las fechas en que el sistema y el medidor detectaron el evento, los tipos de eventos que se deben considerar como mínimo son los siguientes:

- Conexión/Desconexión por puerto óptico.
- Conexión/Desconexión con base de demanda.
- Reset de demanda a través del dispositivo del medidor.
- Activación del modo de prueba.
- Batería baja.
- Flujo de energía inverso con respecto al flujo normal.
- Errores del medidor.
- Ausencia de tensión.
- Restablecimiento de tensión.
- Bajo voltaje.
- Sobre voltaje.
- Comunicación con el puerto óptico.
- Intento para el acceso con password incorrecto.
- Sincronía de tiempo.
- Cambio de estación.
- Cambio de horario.

3.2. - Primer módulo entregable “Login con C.R.U.D” del sistema HES.

Para el primer módulo entregable del sistema Head End System se es necesario crear un login de usuario con JSP, un CRUD (Create, Read, Update and Delete) y una base de datos de usuarios para validación y acceso a sistema.

Tabla 3. 1 Historia de usuario número 1 "Creación Login"(Elaboración propia).

Historia de usuario	Historia de usuario
Id	H1
Nombre	Creación de Login con CRUD.
Prioridad	Media
Riesgo	Medio
Descripción	Creación de un módulo login con JSP para la interfaz gráfica de usuario y Creación de un CRUD con validación de usuario y contraseña en base de datos con Oracle y generada en Hibernate.
Validación	Validar un usuario y contraseña para ingresar a la primera parte del sistema HES, Creación de un usuario con su respectiva contraseña en base de datos.

3.2.1. - Diagramas del primer módulo “Login con CRUD”.

3.2.1.1. - Diagrama de caso de uso “Login con CRUD”.

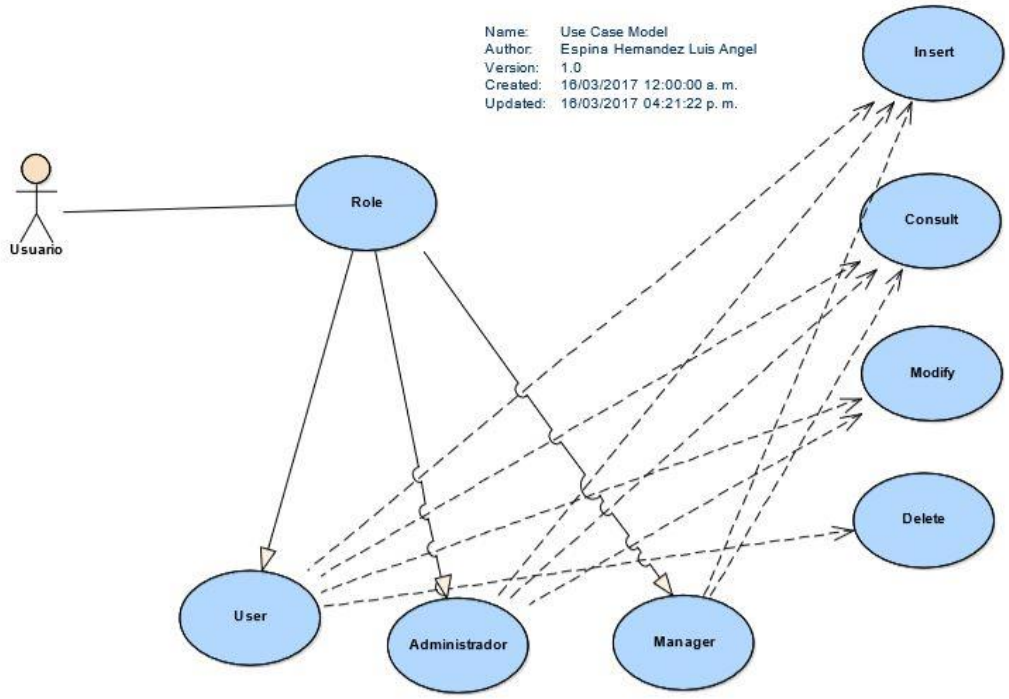


Ilustración 3. 1 Diagrama de caso de uso de un login con CRUD (Elaboración propia).

En la ilustración 3.1 Se muestra el diagrama de caso de uso para un sistema CRUD, donde al usuario se le asigna un rol, dependiendo del rol con el que se cuente podrá hacer uso de las diferentes funciones en el sistema, cuando el rol es Admin, podrá hacer uso de inserción, consultas, modificaciones, y borrado de cualquier registro, en caso de ser administrador, se tendrá acceso a todo menos al borrado de usuarios, para el usuario solo podrá insertar y consultar.

3.2.1.2. Diagrama de estado “Login con CRUD”.

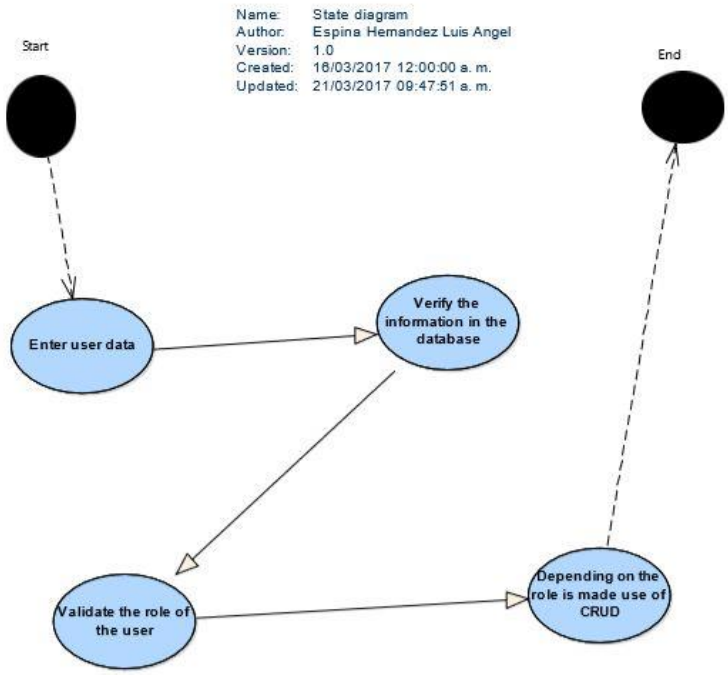


Ilustración 3. 2 Diagrama de estado "Login con CRUD" (Elaboración propia)

En la ilustración 3.2 se muestra el diagrama de estado, donde se muestran los pasos seguidos por el programa, como primer paso el usuario ingresa la información requerida, una vez ingresados se verifican en base de datos para verificar la información, se verifica el rol asignado para que se muestren las partes a las que puede acceder.

3.2.1.3. - Diagrama de flujo “Login con CRUD”.

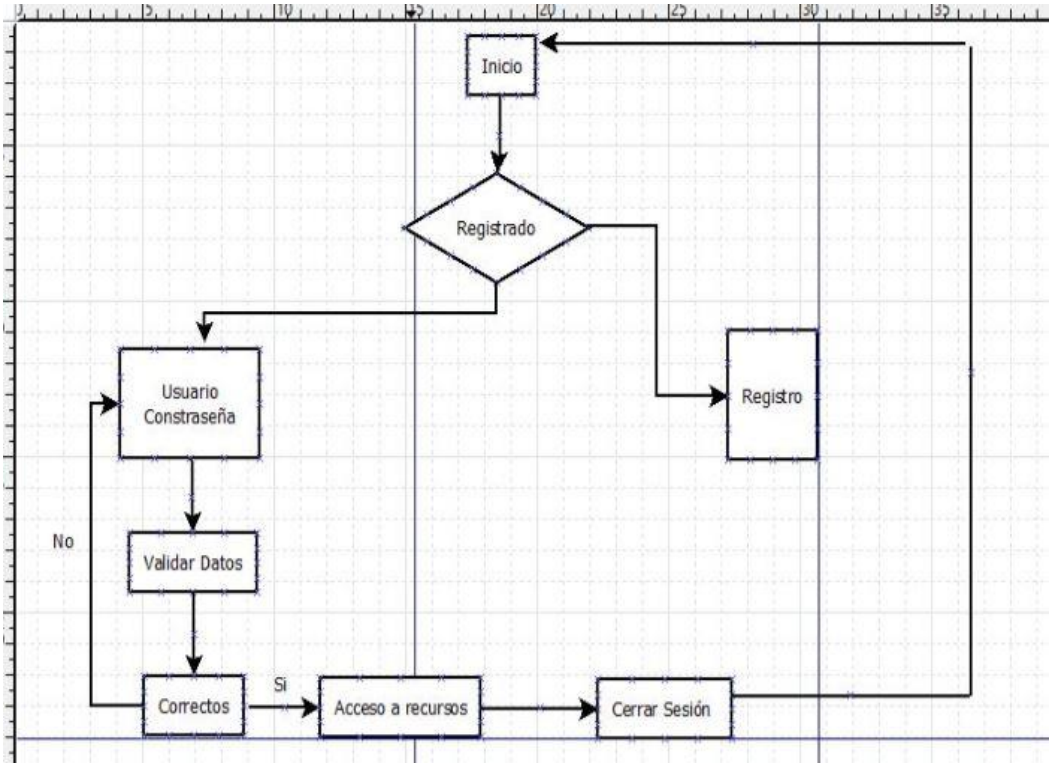


Ilustración 3. 3 Diagrama de flujo Login con CRUD (Elaboracón propia).

En la ilustración 3.3 me muestra el diagrama de flujo de login, para ello se ingresa al sistema si no se tiene un usuario es necesario notificar al administrador para que sea registrado y se le asigne un rol, si es un usuario ya registrado, se verifican los datos de inicio de sesión para el acceso a los recursos del sistema.

3.2.1.4. - Diagrama de secuencia de Login con CRUD.

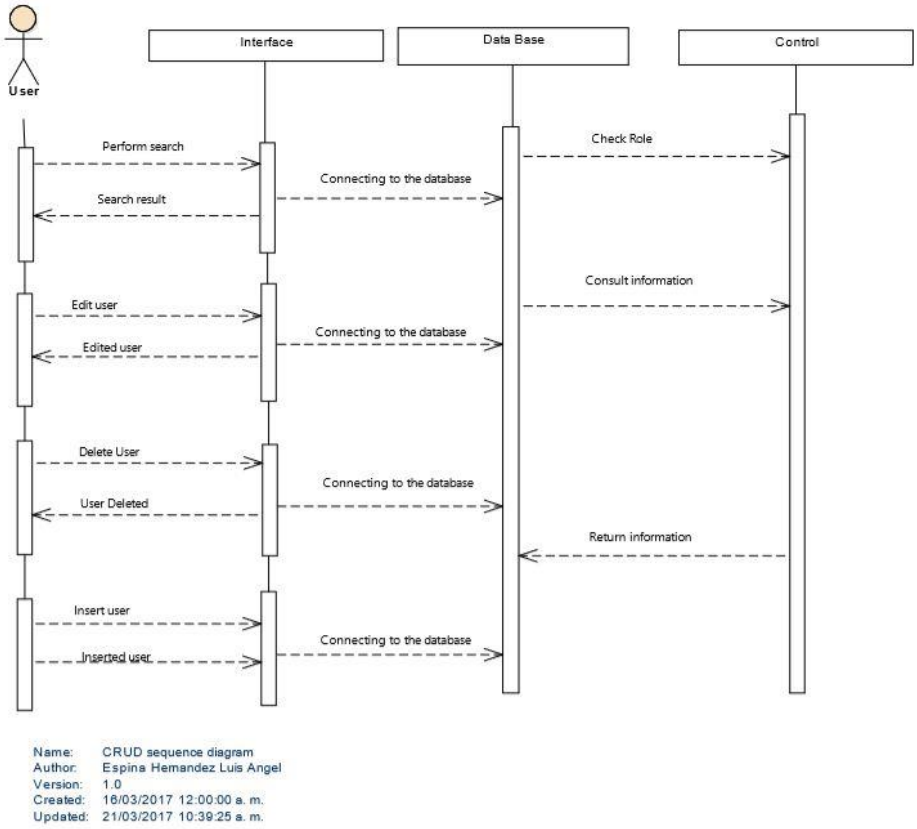


Ilustración 3. 4 Diagrama de secuencia de Login con CRUD (Elaboración propia).

En la ilustración 17 se muestra el diagrama de secuencia, en la cual se muestra el contacto que tiene el usuario en el sistema, pasando por la interfaz gráfica, por un control y una base de datos, básicamente el usuario entra a una interfaz se conecta con la base de datos, se verifica la información de usuario, y regresa la información solicitada, cuando se verifica el rol se da acceso al usuario para que pueda acceder a insertar, consultar, borrar y editar información.

3.2.2. - Tareas para el primer módulo “Login con CRUD”.

Tabla 3. 2 T01 "Creación de páginas web con JSP" (Elaboración propia).

Tarea	T01
Historia de usuario	H1
Estado	Finalizada.
Descripción	Creación De Índex JSP, página de inicio y registro.

Tabla 3. 3 T02 "Creación de Login.jsp" (Elaboración propia).

Tarea	T02
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación Login.jsp, pagina para usuarios registrados.

Tabla 3. 4 T03 "Creación de LogOut.jsp" (Elaboración propia).

Tarea	T03
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de LogOut.jsp, página de cierre de sesión.

Tabla 3. 5 T04 "Creación de página de Registro.jsp" (Elaboración propia).

Tarea	T04
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de Registro.jsp, pagina de registro para usuarios nuevos.

Tabla 3. 6 T05 "Creación de Success.jsp" (Elaboración propia).

Tarea	T05
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de Success.jsp, página de inicio para usuarios ya autentificados.

Tabla 3. 7 T06 "Creación de Welcome.jsp" (Elaboración propia).

Tarea	T06
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de Welcome.jps

Tabla 3. 8 T07 "Configuración del archivo web.xml" (Elaboración propia).

Tarea	T07
Historia de usuario	H1
Estado	Finalizada
Descripción	Configuración del Archivo web.xml

Tabla 3. 9 T08 "Configuración del archivo Pom.xml" (Elaboración propia).

Tarea	T08
Historia de usuario	H1
Estado	Finalizada
Descripción	Configuración del archivo Pom.xml

Tabla 3. 10 T09 "Creación de estación de trabajo en Oracle" (Elaboración propia).

Tarea	T09
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de estación de trabajo en Oracle, se crea un WorkSpace para alojar la base de datos

Tabla 3. 11 T10 "Creación de la tabla Users" (Elaboración propia).

Tarea	T10
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de tabla Users, para el guardado de usuarios.

Tabla 3. 12 T11 "Creación de la tabla Roles" (Elaboración propia).

Tarea	T11
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de tabla Roles, para asignar privilegios a usuarios.

Tabla 3. 13 T12 "Creación de llaves primarias." (Elaboración propia).

Tarea	T12
Historia de usuario	H1
Estado	Finalizada
Descripción	Creación de llaves Primarias, para la relación entre tablas.

3.3. - Segundo modulo entregable “URL Unica” del sistema HES.

El segundo módulo entregable del sistema HES, es desarrollar un módulo que trabaja en conjunto con la interfaz gráfica con un patrón mvc, es decir cada que el usuario envía una solicitud en el navegador web se pone en marcha, pasa por el Dispatcher de spring MVC el cual protege al sistema de ataques de inyección de código, así que se muestra una URL única que no muestra en que parte del sistema se encuentra el usuario.

Tabla 3. 14 “Historia de Usuario Número 2” (Elaboración propia).

Historia de usuario	Historia de usuario
Id	H2
Nombre	Creación Url´s únicas con spring MVC
Prioridad	Media
Riesgo	Medio
Descripción	Crear un módulo que proteja el sistema para evitar inyecciones de código mostrando una url única para todo el sistema.
Validación	Implementación a clase login.

3.3.1. - Diagramas del segundo modulo “URL Única”.

3.3.1.1. - Diagrama de caso de uso “URL Única”.

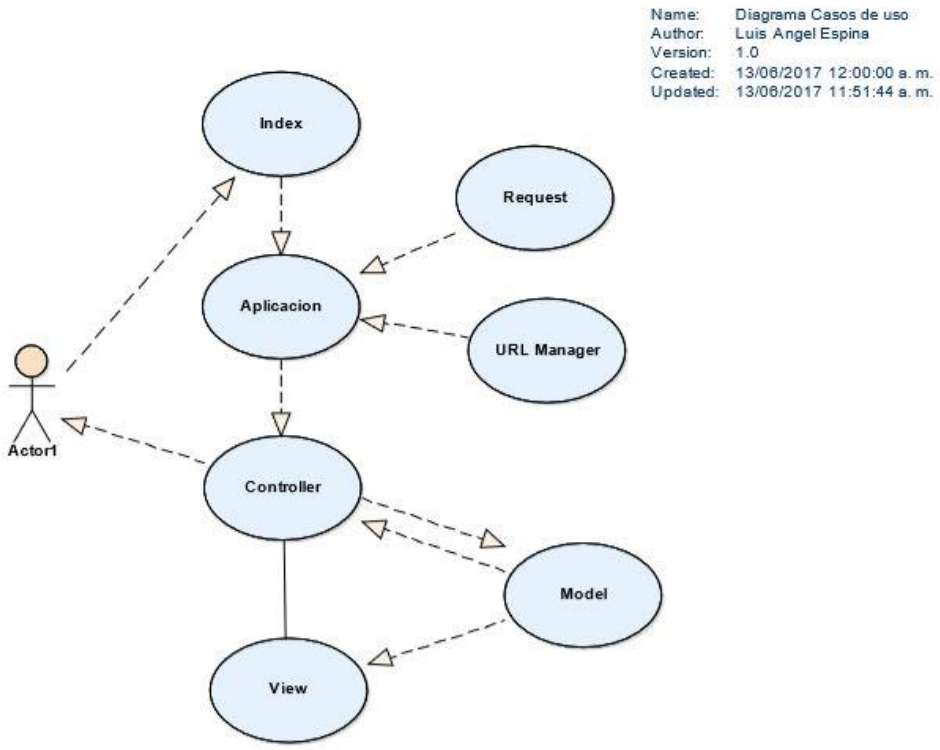


Ilustración 3. 5 Diagrama de Caso de uso "URL Unica" (Elaboración propia).

En la ilustración 3.5 se muestra que cuando el usuario entra al sistema e inicia a interactuar la solicitud abandona el navegador y lleva información que el usuario necesita, la solicitud va a incluir la URL solicitada, así que entra al URL controlador que es un componente de spring que procesa la solicitud para mostrar el contenido en el JSP.

3.3.1.2. - Diagrama de clases “URL Única”.

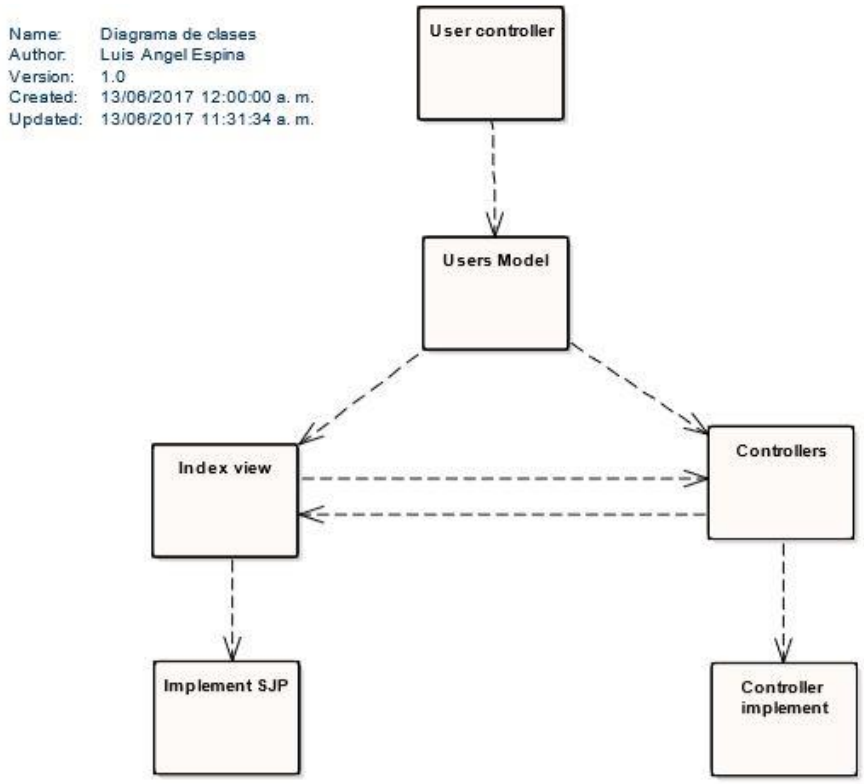
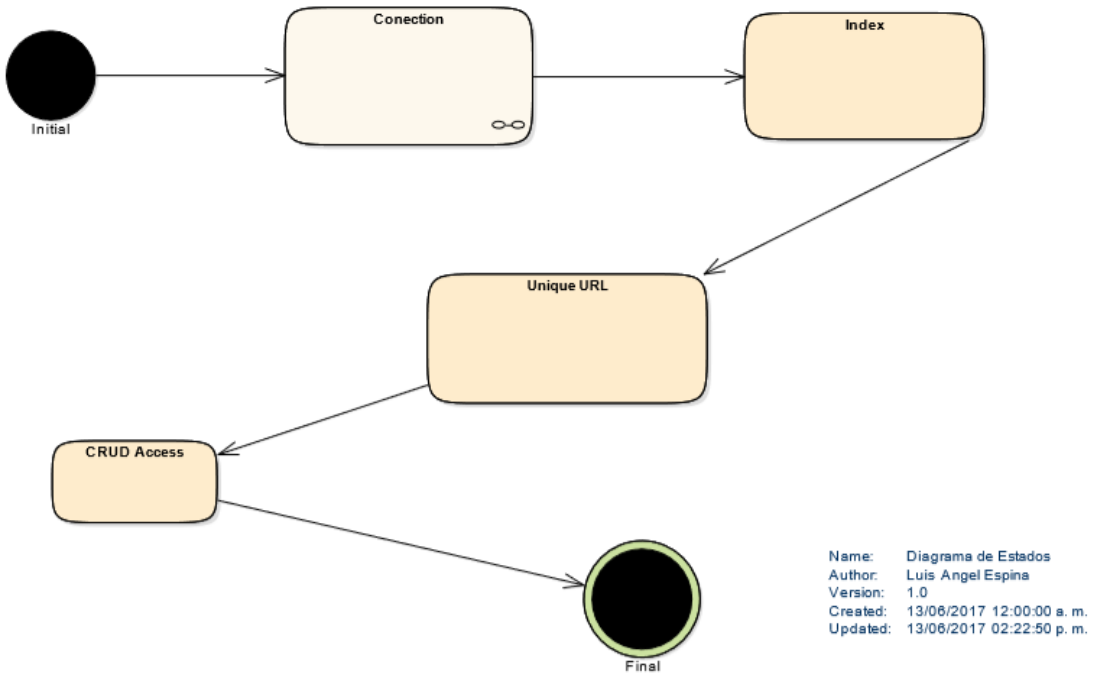


Ilustración 3. 6 Diagrama de clases "URL Única" (Elaboración propia).

En la ilustración 3.6 se muestran que el modulo está compuesto por la clase de entities, que es la clase que genera la base de datos para usuarios, los controllers están compuestos por la clase UserServicesImpl, Userservices y la clase UserDaoImpl que son los que controlan todo el acceso así como las restricciones.

3.3.1.3. - Diagrama de estado “URL Única”.



Única

Ilustración 3. 7 Diagrama de estado "URL única" (Elaboración propia).

En la ilustración 3.7 se muestra el diagrama de estado donde se observa a grandes rasgos los puntos por los que pasa el usuario, cuando hay una conexión con la base de datos se pasa a un índex que al enviar una solicitud para mostrar el sistema CRUD con una URL única para no ver en qué parte del sistema se encuentra el usuario.

3.3.1.4. - Diagrama de Flujo “URL Única”.

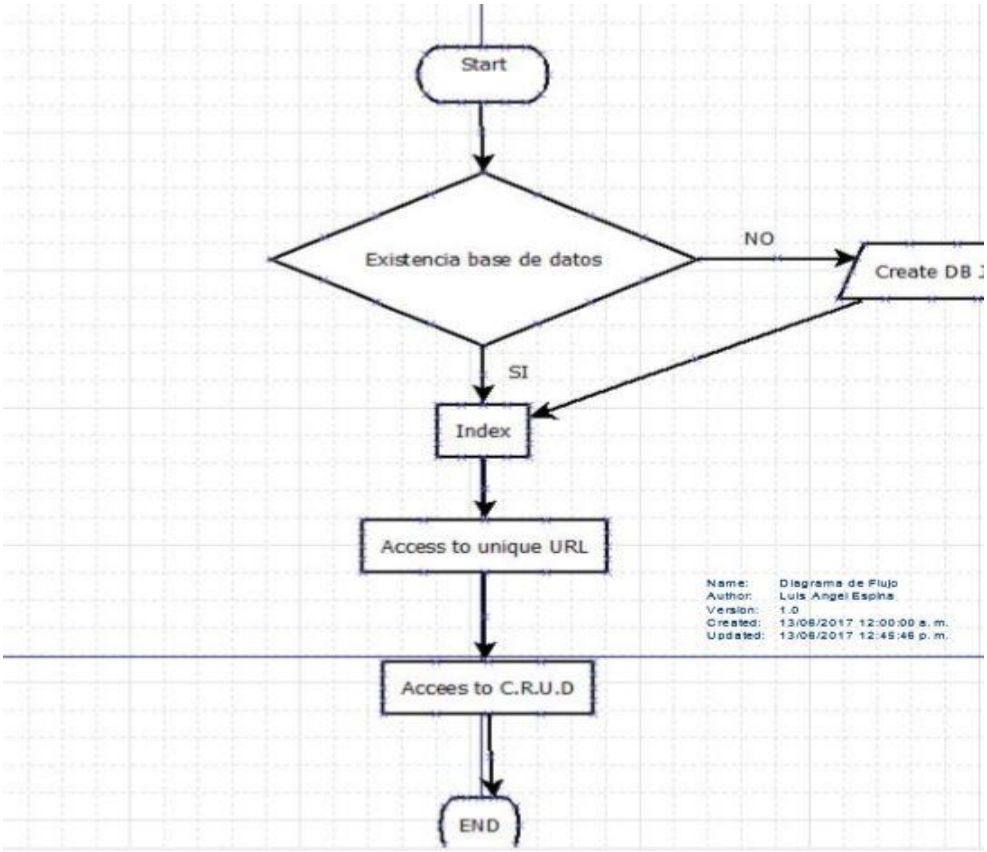


Ilustración 3. 8 Diagrama de flujo "URL Única" (Elaboración propia).

En la ilustración 3.8 Se muestra el diagrama de flujo del módulo de URL única, cuando el usuario es existente en base de datos, se muestra un índex de inicio, y se accede al sistema del CRUD con una dirección única.

3.3.1.5. - Diagrama de secuencia “URL Única”.

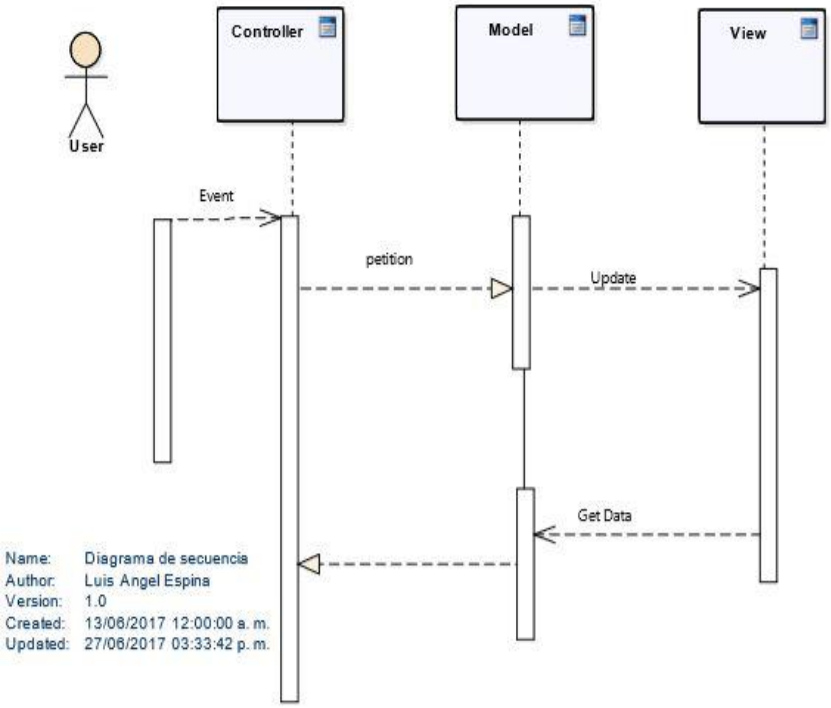


Ilustración 3. 9 Diagrama de secuencia "URL única" (Elaboración propia).

En la ilustración 3.9 Se muestra un diagrama de secuencia donde el usuario envía una petición hacia el modelo y recibe la información, pasando por todas las vistas del sistema pero con una sola dirección.

3.3.2. - Tareas para el segundo modulo “URL Única”.

Tabla 3. 15 T01 "Creación Estación de trabajo" (Elaboración propia).

Tarea	T01
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación de Espacio de trabajo en Eclipsé.

Tabla 3. 16 T02 "Agregar dependencias a maven" (Elaboración propia).

Tarea	T02
Historia de usuario	H2
Estado	Finalizada
Descripción	Agregar dependencias al archivo Pom de maven para dependencias necesarias.

Tabla 3. 17 T03 "Configuración de Web.xml" (Elaboración propia).

Tarea	T03
Historia de usuario	H2
Estado	Finalizada
Descripción	Configuración del archivo web.xml para la declaración y mapeo.

Tabla 3. 18 T04 "Creación de Hello.jsp" (Elaboración propia).

Tarea	T04
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación de la página Hello.jsp

Tabla 3. 19 T05 "Configuración del Archivo Application Context" (Elaboración propia).

Tarea	T05
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación y configuración del Archivo Applicaioncontext.xml

Tabla 3. 20 T06 "Configuración Archivo SpringMVC-Servlet" (Elaboración propia).

Tarea	T06
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación y configuración del archivo SpringMvc-servlet.xml

Tabla 3. 21 T07 "Creacion de Index.JSP" (Elaboración propia).

Tarea	T07
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación de la página Index.jsp

Tabla 3. 22 T8 "Creación Del Paquete Com.Controllers" (Elaboración propia).

Tarea	T08
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación del paquete com.Controllers y la clase UserControler.

Tabla 3. 23 T09 "Creación del paquete Com.DAO" (Elaboración propia).

Tarea	T09
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación del paquete com.Dao y la clase UserDao

Tabla 3. 24 T10 "Creación Del Paquete Com.Services" (Elaboración propia).

Tarea	T10
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación del paquete Com.DaoImpl y la clase UserDaoImpl

Tabla 3. 25 T11 "Creación Del Paquete Com.Services" (Elaboración propia).

Tarea	T11
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación del paquete Com.Services y la clase UserServices

Tabla 3. 26 T12 "Creación Del Paquete Com.ServicesImpl" (Elaboración propia).

Tarea	T12
Historia de usuario	H2
Estado	Finalizada
Descripción	Creación del paquete Com.ServicesImpl y la clase UserServicesImpl

3.4. - Tercer módulo entregable “Base de datos relacional” para el sistema HES.

En este Sprint se realiza la descripción del modelo entidad relación necesario para desarrollar y explotar el sistema HES. Tal y como se describe en este Sprint la base de datos relacional será desarrollada con JPA (Java Persistence API) e Hibernate y como sistema gestor Oracle, en las entidades que permiten representar la información almacenada para que dicha información sea utilizada por la lógica de negocios del sistema, por otro lado tambien se diseñan tablas para el crecimiento del sistema y que pueda ser utilizado por cualquier tipo de Utility y cualquier sistema gestor de base de datos que requeira.

Tabla 3. 27 H3 "Modelación y creación de la base de datos relacional" (Elaboración propia).

Historia de usuario	Historia de usuario
Id	H3
Nombre	Modelación y Creación de la Base de datos Relacional.
Prioridad	Alta
Riesgo	Alto
Descripción	Crear la base de datos relacional de cada una de las entidades que requiere el sistema, así como sus llaves primarias y foráneas, con sus respectivas relaciones en cada entidad, así como las clases DAO, los constrains de cada una de las entidades, además de los Setters y Getters de cada clase.
Validación	Insertar Un registro en cada entidad.

3.4.1. - Modelo entidad relación de la base de datos del sistema HES.

El modelo entidad relación para la base de datos tiene como principal funcionalidad guardar el registro de las actividades realizadas por la red de medidores así como las dependencias de los dispositivos. Cabe resaltar que CFE cuenta con sus propias bases de datos donde se guarda la información propia para que realice sus procesos de facturación, debido a la naturaleza innovadora del proyecto se encontró la necesidad de crear un modelo donde se ubicaran las asociaciones entre dispositivos; por ejemplo los medidores que están relacionados con un gabinete, y todos los dispositivos hacia el usuario al cual le corresponde el servicio.

Esta base de datos fue cambiando a lo largo del desarrollo del proyecto agregando tablas, eliminando campos y alteando a estos, debido al mejoramiento continuo del proyecto.

Tras haber realizado el análisis de requerimientos por el equipo desarrollador se modelo un diagrama de Entidad-Relacion el cual se muestra en la ilustración 3.10.

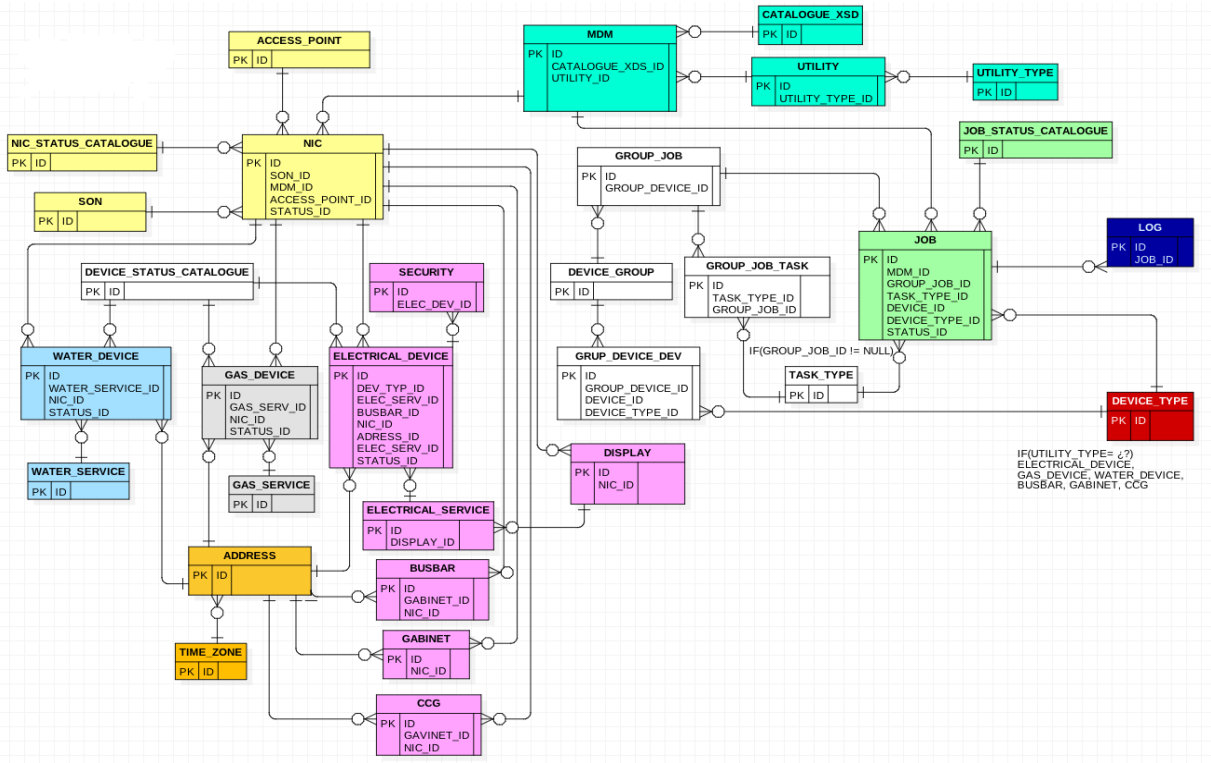


Ilustración 3. 10 Modelo entidad relación sistema HES (Elaboración propia).

3.4.2. - Tareas a realizar para el módulo 3 “Base de datos relacional”.

Tabla 3. 28 T01 "Entidad Gas_Service y sus Atributos" (Elaboración propia).

Tarea	T01
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Gas_Service, crear la secuencia para la llave primaria, relación @OneToMany con entidad Gas_Device, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 29 T02 "Entidad Gas_Device y sus atributos" (Elaboración propia).

Tarea	T02
Historia de usuario	H3
Estado	Finalizada.
Descripción	Creación de la entidad Gas_Device, crear su secuencia para auto incremento de llave primaria, crear las relación @ManyToOne con la entidad Address, @ManyToOne con la entidad Gas_service, @ManyToOne con DeviceStatusCatalogue, @ManyToOne con NIC, @ManyToOne con Address, crear constructor sin argumentos de tipo público, crear un constructor con métodos getter y setter asociados, creación del método ToString y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 30 T03 "Entidad Electrical_Service y sus atributos" (Elaboración propia).

Tarea	T03
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Electrical_Service, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con entidad Electrical_Service, @ManyToOne con la entidad Address, @ManyToOne con la entidad Device_Status_Catalogue, @ManyToOne con la entidad security, @ManyToOne con la entidad NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToSting para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 31 T04 "Entidad Electrical_Device y sus atributos" (Elaboración propia).

Tarea	T04
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Electrical_Device, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con entidad Security, @ManyToOne con la entidad Electrical_Service, @ManyToOne con la entidad Address, @ManyToOne con la entidad Device_Status_Catalogue, @ManyToOne con la entidad NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 32 T05 "Entidad Security y sus atributos" (Elaboración propia).

Tarea	T05
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Security, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Electrical_Device, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 33 T06 "Entidad BusBar y sus atributos" (Elaboración propia).

Tarea	T06
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad BusBar, crear la secuencia de auto incremento para la llave primaria, con relación @ManyToOne con la entidad Address, @ManyToOne con la entidad Device_Status_Catalogue, @ManyToOne con la entidad NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 34 T07 "Entidad Cabinet y sus atributos" (Elaboración propia) (Elaboración propia).

Tarea	T07
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Cabinet, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Address, @ManyToOne con la entidad Device_Status_Catalogue, @ManyToOne con la entidad NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 35 T08 "Entidad CCG y sus atributos" (Elaboración propia).

Tarea	T08
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad CCG, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Address, @ManyToOne con la entidad Device_Status_Catalogue, @ManyToOne con la entidad NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 36 T09 "Entidad Address y sus atributos" (Elaboración propia).

Tarea	T09
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Address, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad BusBar, @OneToMany con la entidad Cabinet, @OneToMany con la entidad CCG, @OneToMany con la entidad Electrical_Device, @OneToMany con la entidad Gas_Device, @OneToMany con la entidad Water_Device y @ManyToOne con la entidad Time_zone, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 37 T10 "Entidad Time_Zone y sus atributos" (Elaboración propia).

Tarea	T10
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Time_Zone, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad Address, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 38 T11 "Entidad Water_Device y sus atributos" (Elaboración propia).

Tarea	T11
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Water_Device, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Address, @ManyToOne con la entidad Device_Status_Catalogue, @ManyToOne con la entidad NIC, @ManyToOne con Water_Service crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 39 T12 "Entidad Device_Status_Catalogue y sus atributos" (Elaboración propia).

Tarea	T12
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Device_Status_Catalogue, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad Water_Device, @OneToMany con la entidad Gas_Device, @OneToMany con la entidad Electrical_Device, @OneToMany con la entidad BusBar, @OneToMany con la entidad Cabinet, @OneToMany con la entidad CCG, @OneToMany con la entidad Display, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 40 T13 "Entidad Display y sus atributos" (Elaboración propia).

Tarea	T13
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Display, crear la secuencia de auto incremento para la llave primaria, Crear relación, @ManyToOne con la entidad Device_Status_Catalogue, @ManyToOne con la entidad NIC, @OneToMany con Electrical_Service crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 41 T14 "Entidad Water_Service y sus atributos" (Elaboración propia).

Tarea	T14
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Water_Service, crear la secuencia de auto incremento para la llave primaria, Crear relación, @OneToMany con Water_Device, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 42 T15 "Entidad Group_Job y sus atributos" (Elaboración propia).

Tarea	T15
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Group_Job, crear la secuencia de auto incremento para la llave primaria, Crear relación, @OneToMany con la entidad Group_Job_Relationship, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 43 T16 "Entidad Utility y sus atributos"(Elaboración propia).

Tarea	T16
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Utility, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Utility_Type, @OneToMany con la entidad MDM crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 44 T17 "Entidad Access_Point y sus atributos" (Elaboración propia).

Tarea	T17
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Access_Point, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 45 T18 "Entidad Nic_Status_Catalogue y sus atributos" (Elaboración propia).

Tarea	T18
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Nic_Status_Catalogue, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 46 T19 "Entidad SON y sus atributos" (Elaboración propia).

Tarea	T19
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad SON, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con NIC, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 47 T20 "Utility_Type y sus atributos" (Elaboración propia).

Tarea	T20
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Utility_Type, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad Utility, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 48 T21 "Entidad NIC y sus atributos" (Elaboración propia).

Tarea	T21
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad NIC, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Nic_Status_Catalogue, @ManyToOne con la entidad Access_Point, @ManyToOne con la entidad MDM, @ManyToOne con la entidad SON, @OneToMany Con la entidad Water_Device, @OneToMany con la entidad Gas_Device, @OneToMany con la entidad Electrical_Device, @OneToMany con la entidad Display, @OneToMany con la entidad BusBar, @OneToMany con la entidad Cabinet, @OneToMany con la entidad CCG, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 49 T22 "Entidad MDM y sus atributos" (Elaboración propia).

Tarea	T22
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad MDM, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad NIC, @ManyToOne con la entidad Catalogue_XSD, @ManyToOne con la entidad Utility, @ManyToOne con la entidad JOB, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 50 T23 "Entidad Catalogue_XSD y sus atributos" (Elaboración propia).

Tarea	T23
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Catalogue_XSD, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad MDM, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 51 T24 "Entidad Log y sus atributos" (Elaboración propia).

Tarea	T24
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Log, Crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad Group_Job_Relationship, @ManyToOne con la entidad MDM, @OneToMany con la entidad Group_Job, @OneToMany Job_Dev_Relationship, @OneToMany con la entidad JOB, @ManyToOne con la entidad Task_Type, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 52 T25 "Entidad Job_Status_Catalogue y sus atributos" (Elaboración propia).

Tarea	T25
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Job_Status_Catalogue, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad MDM, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 53 T26 "Entidad Job y sus atributos" (Elaboración propia).

Tarea	T26
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Job, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad LOG, @ManyToOne con la entidad JosStatusCatalogue, @ManyToOne con la entidad Task_Type, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 54 T27 "Entidad Task_Type y sus atributos" (Elaboración propia).

Tarea	T27
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Task_Type, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad JOB, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 55 T28 "Entidad Device_Group y sus atributos" (Elaboración propia).

Tarea	T28
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Device_Group, crear la secuencia de auto incremento para la llave primaria, Crear relación @OneToMany con la entidad Job_Dev_Relationship, @OneToMany con la entidad Device_Group, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 56 T29 "Entidad Device_Group_Relationship y sus atributos" (Elaboración propia).

Tarea	T29
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Device_Group_Relationship, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Device_Group, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 57 T30 "Entidad Job_Dev_Relationship y sus atributos" (Elaboración propia).

Tarea	T30
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Job_Dev_Relationship, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad Device_Group, @ManyToOne con la entidad JOB, @ManyToOne con la entidad Group_JOB, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

Tabla 3. 58 T31 "Entidad Group_Job_Relationship y sus atributos" (Elaboración propia).

Tarea	T31
Historia de usuario	H3
Estado	Finalizada.
Descripción	Crear la entidad Gruop_Job_Relationship, crear la secuencia de auto incremento para la llave primaria, Crear relación @ManyToOne con la entidad JOB, @ManyToOne con la entidad Group_Job, crear constructor sin argumentos de tipo público, crear un constructor con métodos setter y getter asociados, creación de método ToString para consultas y creación de las clases DAO (Objeto de acceso a datos) y DAOImpl.

3.5. - Cuarto modulo entregable “Spring securtiy” para el sistema HES.

El cuarto modulo del sistema HES es el encargado de la seguridad del sistema con el framework de Spring Security, con el cual se controla el acceso a la información del sistema, dando privilegios a cada uno de ellos.

Tabla 3. 59 Historia de usuario para el cuarto modulo “Spring securtiy” (Elaboración propia).

Historia de usuario	Historia de usuario
Id	H4
Nombre	Modelación y Creación de la Base de datos para roles y usuarios y creación de los archivos XML de control.
Prioridad	Media
Riesgo	Media
Descripción	Módulo de seguridad que controle accesos a información y asignación de privilegios.
Validación	Crear un usuario con cada rol.

3.5.1. - Diagramas para el módulo 4 “Spring securtiy” del sistema HES.

3.5.1.1. - Diagrama de secuencia de “Spring securtiy”.

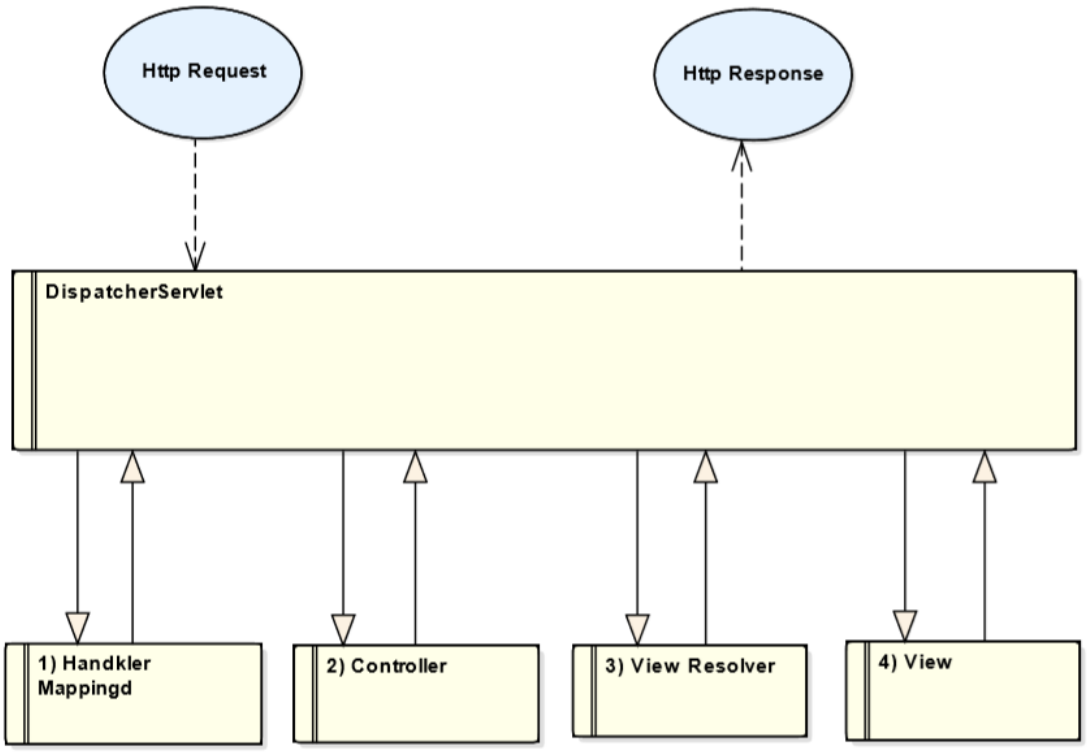


Ilustración 3. 11 Diagrama de secuencia de Spring Security (Creación propia).

En la ilustración 3.11 se muestra la secuencia de una solicitud en Spring Security, donde después de recibir una solicitud Http, el DispatcherServlet consulta en el HandlerMapping para llamar al controlador apropiado para la seguridad, el controlador toma la solicitud y llama a los métodos de servicio apropiados en función del método GET o POST utilizados donde la lógica define y devuelve el nombre de la vista a DispatcherServlet y este a su vez toma la ayuda de la viewResolver para recoger la vista definida para la solicitud y una vez que finaliza la vista el DispatcherServlet pasa los datos del modelo a la vista que finalmente se procesa en el navegador.

3.5.1.2. - Diagrama de caso de uso de “Spring securtiy”.

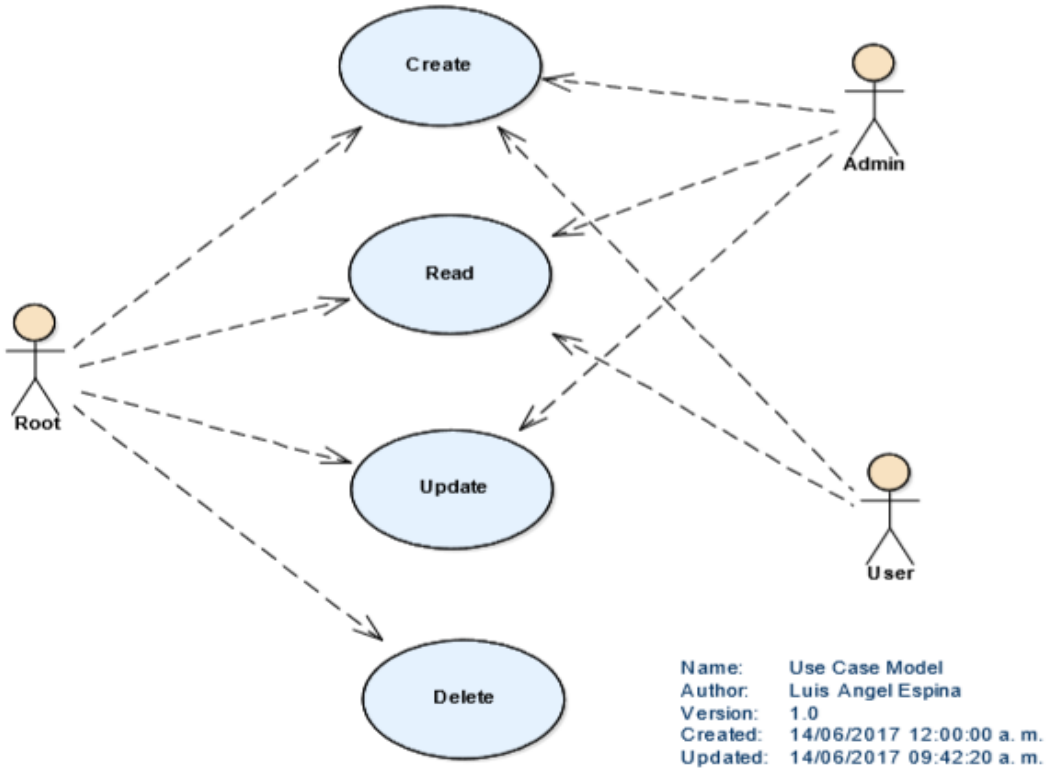


Ilustración 3. 12 Diagrama de caso de uso de Spring Security (Creación propia).

En la ilustración 3.12 se muestra el diagrama de caso de Spring Security, la cual muestra los principales roles de usuario (Root, Admin y usuario) ya que de rol que sea asignado a cada usuario podrá acceder a la información de cada vista que solicite, donde el Root tendrá acceso a todas las vistas y acceso a crear, editar, consultar y borrar, el administrador solo a crear, consultar y actualizar y por último el usuario solo podrá crear y actualizar,

3.5.1.3. - Modelo E-R de Spring Security.

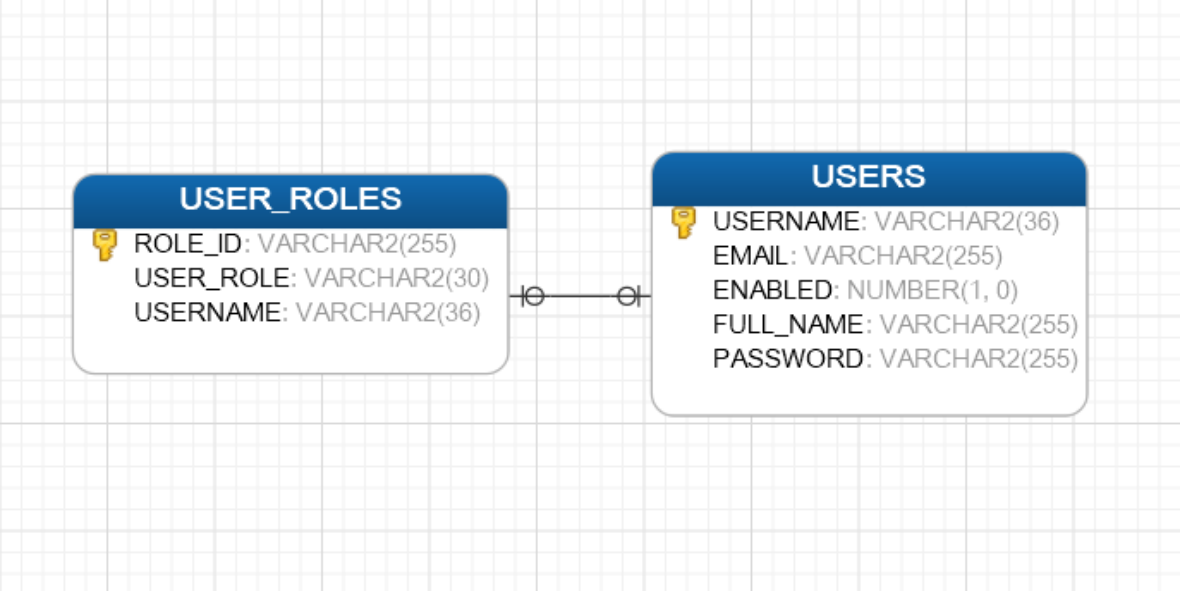


Ilustración 3. 13 Modelo entidad-relación Spring Security (Elaboración propia).

En la ilustración 57 se muestra el modelo entidad relación donde Spring Security se basa para los roles de usuario, la entidad Users contiene una llave primaria llamada UserName de tipo número y auto incrementable, los campos E-Mail, Enabled, Full_Name y Password son te tipo varchar. En la entidad User_Roles se tiene una llave primaria de tipo número y auto incrementable, un User_Role (Role_User, Role_Admin, Role_Root) de tipo varchar y una llave foránea llamada UserName con una relación @ManyToOne con la entidad Users.

3.5.2. - Tareas a realizar para el cuarto modulo “Spring securtiy”.

Tabla 3. 60 T01 "Creación estación de trabajo".

Tarea	T01
Historia de usuario	H4
Estado	Finalizada
Descripción	Crear Espacio de trabajo para la elaboración del módulo.

Tabla 3. 61 T02 "Creación del archivo Pom.xml".

Tarea	T02
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación y configuración del archivo Pom.xml

Tabla 3. 62 T03 "Archivo Web.xml".

Tarea	T03
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación y configuración del archivo Web.xml

Tabla 3. 63 T04 "Archivo Spring.security.xml".

Tarea	T04
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación y configuración del archivo Spring-security.xml

Tabla 3. 64 T05 "Archivo Spring-DataBase.xml".

Tarea	T05
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación y configuración del archivo Spring-DataBase.xml

Tabla 3. 65 T06 "Archivo MVC-Dispatcher-Servlet.xml".

Tarea	T06
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación y configuración del archivo MVC-Dispatcher_Servlet.xml

Tabla 3. 66 T07 "Archivo 403.jsp".

.Tarea	T07
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación del archivo de la página de prueba 403.jsp

Tabla 3. 67 T08 "Archivo Admin.jsp".

Tarea	T08
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación del archivo de prueba Admin.jsp

Tabla 3. 68 T09 "Archivo Login.jsp".

Tarea	T09
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación del archivo de prueba Login.jsp

Tabla 3. 69 T10 "Archivo Registration.jsp".

Tarea	T10
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación del archivo de prueba Registration.Jsp

Tabla 3. 70 T11 "Archivo Welcome.jsp".

Tarea	T11
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación del archivo de prueba Welcome.sjp

Tabla 3. 71 T12 "Paquete Dal.Security".

Tarea	T12
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación del paquete de Dal.Security

Tabla 3. 72 T13 "Archivo Conexion.DB.java".

Tarea	T13
Historia de usuario	H4
Estado	Finalizada
Descripción	Crear archivo de Conexión.DB.java

Tabla 3. 73 T14 "Archivo Main.Controller.java".

Tarea	T14
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación de la clase Main.Controller.java

Tabla 3. 74 T15 "Archivo Test.Java".

Tarea	T15
Historia de usuario	H4
Estado	Finalizada
Descripción	Creación de Test.Java

Capítulo 4. Pruebas y resultados.

En el presente capítulo se muestran imágenes de la interfaz del Head End System en un entorno de pruebas, resultado del desarrollo alcanzado por el equipo.

La base de datos y el sistema Spring Security fueron ejecutados en Oracle DataBase 11g en un equipo con 12 Gb de memoria RAM, y un microprocesador de 8 núcleos a 2.3 GHz corriendo en un sistema operativo Windows 10. Para la visualización de las tablas se utilizó el navegador Web Google Chrome. En la fase de desarrollo que se tuvo, el equipo de desarrollo se hizo un uso aleatorio de registros en la base de datos para trabajar con el sistema Head End System. Para las pruebas de tareas a medidores se utilizó un gabinete de 4 medidores.

En esta sección se muestran los resultados de cada una de las tareas de cada historia de usuario asi como una breve descripción de lo mostrado.

4.1. - Resultados primer módulo llamado Login con C.R.U.D.

Ilustración 4. 1 "Index.JSP T1 primer módulo (Resultado)".

En la ilustración 4.1 se muestra el resultado del archivo index.jsp el cual es un login a nivel base de datos, cuenta con el nombre del usuario y su respectivo Password, con un link hacia una pagina de registro para usuarios nuevos.

Ilustración 4. 2 "LogOut.jsp T2 de la historia 1 (Resultado)".

En la ilustración 4.2 se muestra el Script que el sistema arroja una vez que el usuario cierra sesión, dicho sript contiene un SetAttribute el cual contiene el UserId y muestra el nombre del usuario con el id correspondiente.

Ilustración 4. 3 "Reg.jsp T3 de la historia 1 (Resultado)".

En la ilustración 4.3 se muestra el resultado de la tarea 3 de la primera historia de usuario, a la cual se puede accesar por medio de la pagina Index.jsp, en Reg.jsp se hace un registro a base de datos con los datos requeridos.

Ilustración 4. 4 "T4 Registration.jsp (Resultado)".

En la ilustración 4.4 se muestra la directive.page básicamente es la biblioteca que utiliza “jsp” para conectarse con la base de datos y el Scriplet contiene las credenciales de acceso a base de datos y la información que se envía hacia ella.

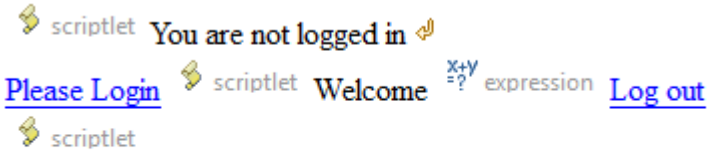


Ilustración 4. 5 T5 “Succes.jsp” (Resultado).

En la ilustración 4.5 se muestra el resultado de la pagina Sussces.jsp la cual contiene un primer Scriptlet el cual muestra la información del usuario que ingreso identificado con su id, en caso de no ingresar de forma correcta arroja “You are not logged in ” y una liga que lo regresa a Index.jsp, el segundo Scriptlet con un comando Else que da la bienvenida al sistema con la Expression que es el nombre del usuario logeado identificado con su id, por ultimo el Scriplet final envía una liga de sierre de sesión.

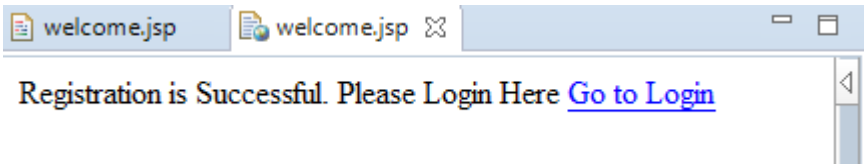


Ilustración 4. 6 T6 "Welcome.jsp" (Resultado).

En la ilustración 4.6 se muestra el resultado de la creacionde Welcome.jsp, donde una vez que el usuario es dado de alta en el sistema, avisa que es satisfactorio y lo direcciona a la pagina de inicio para que pueda ingresar al sistema.

4.2. - Resultados segundo modulo llamado “URL Única”.

Para el segundo módulo del sistema HES estos fueron los resultados de las tareas hechas.

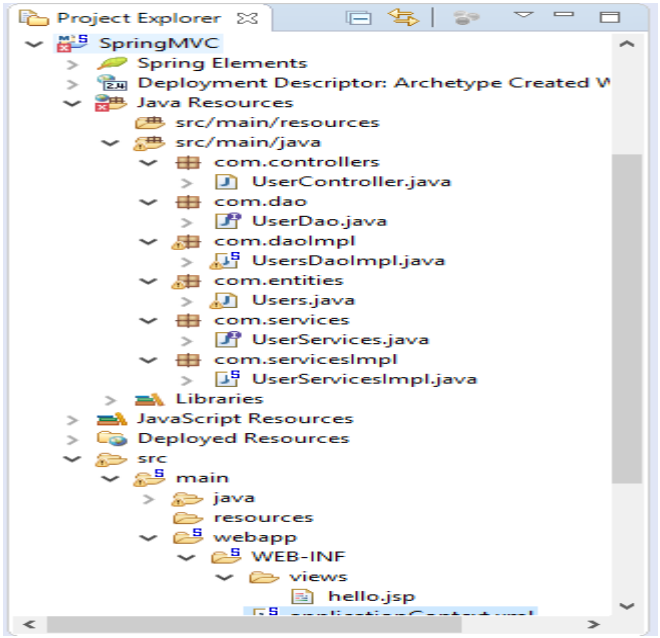


Ilustración 4. 7 T01 del segundo modulo (Elaboración propia).

En la ilustración 4.7 se muestra el resultado de la creación final de la estación de trabajo en eclipce con todos los paquetes y clases que contiene.

Para la T02 seconfigura el archivo POM.xml se encuentran todas las dependencias que se utilizan para el desarrollo de ese modulo.

Para la T03 se configura el archivo web.xml donde se declara el DispatcherServlet para actuar como controlador frontal para manejar todas las solicitudes web.

Para la T04 Se configura el archivo ApplicationContext.xml el cual abre una sesión con spring para todas las transacciones de base de datos.

En la T05 se configura el archivo Springmvc-servlet en el cual se declara el controlador donde se asocia con el viewcontroller lo que significa que este manejara la solicitud. y el ViewResolver define donde spring buscara la página de visualización.

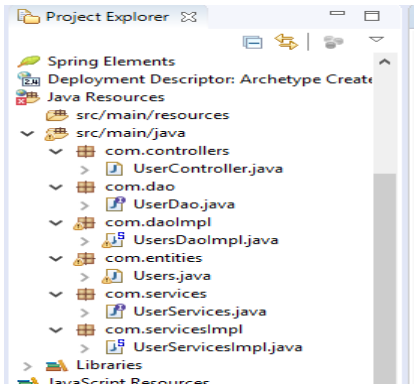


Ilustración 4. 8 Tareas 9, 10, 11 y 12 (Resultado).

En la ilustración 4.8 se muestran las tareas 9 a 12 donde la clase UserController mapea la entidad llamada Users y la principal función de esta es que contiene los métodos para las rutas relativas e indicar que se va a mostrar en cada vista, la clase DAO genera una lista publica de los usuarios que pueden acceder, la clase UsersDAOImpl se inicia sesión con spring para hacer una búsqueda de datos en el modelo, La clase Users genera la tabla de usuarios con Hibernate en Oracle.

4.3. - Resultados del tercer modulo llamado “Base de datos relacional”.

Para el desarrollo del modelo entidad relación se crearon las entidades conforme las relaciones lo pedían ya que unas entidades dependen de otras por ello no se podían crear de manera aleatoria, por lo que se estudiaron las relaciones que se tenían y se crearon en el orden que se muestra en este apartado.

UTILITY_TYPE

	Field	Type	Extra
P	UTILITY_TYPE_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100216	UTILITY_TYPE_ID	Unique

Ilustración 4. 9 Entidad Utility_Type Resultado (Elaboración propia).

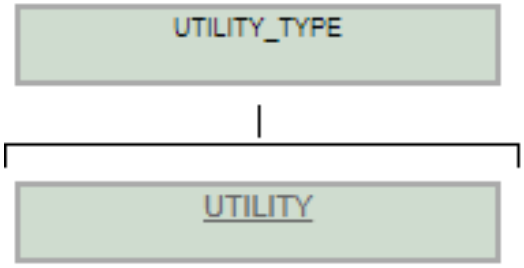


Ilustración 4. 10 Modelo de relación Utility_Type y Utility (Elaboración propia).

En la ilustración 4.9 se muestra la entidad Utility_Type esta tabla está pensada para la escalabilidad del sistema la cual guardaría información del tipo de Utility que se utilizara ya sea agua, gas o energía eléctrica, cuanta con una llave primaria llamada Utility_Type_Id de tipo número y auto incrementable, los campos Description y Name ambos de tipo varchar.

UTILITY

	Field	Type	Extra
P	UTILITY_ID	NUMBER(19)	
	NAME	VARCHAR2(255 CHAR)	Allow Null
	UTILITY_TYPE_ID	NUMBER(19)	Allow Null

Index	Fields	Extra
SYS_C00100214	UTILITY_ID	Unique

Ilustración 4. 11 Entidad Utility Resultado T16 (Elaboración Propia).

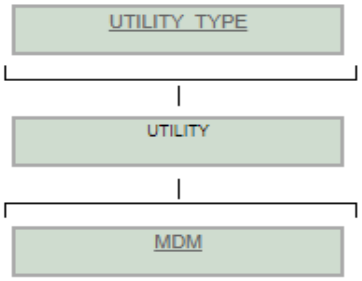


Ilustración 4. 12 Modelo de relación de Utility (Elaboración propia).

En la ilustración 4.11 se muestra la entidad Utility, encargada de guardar la información de la compañía que presta los servicios, ya sea luz, agua, o gas, contiene una llave primaria llamada Utility_Id de tipo número y auto incrementable un campo llamado Name de tipo varchar y una llave foránea llamada Utility_Type_Id

CATALOGUE_XSD

	Field	Type	Extra
P	CATALOGUE_XSD_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100132	CATALOGUE_XSD_ID	Unique

Ilustración 4. 13 Entidad Catalogue_XSD Resultado T23 (Elaboración Propia).

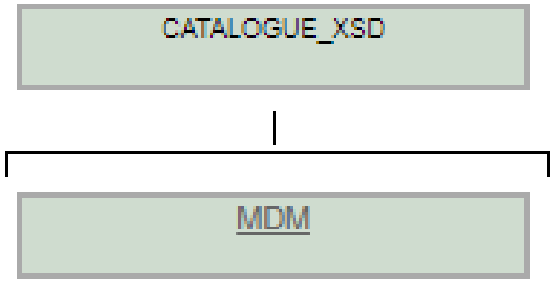


Ilustración 4. 14 Modelo De relación De Catalogue_XSD (Elaboración Propia).

En la ilustración 4.13 se muestra la entidad Catalgoue_XSD en esta tabla se guarda la información de los archivos XSD que se utilizan para la comunicación correcta entre el sistema HES y el sistema MDM donde se encuentran las tareas a realzar, cuenta con una llave primaria llamada Catalogue_XSD_Id De tipo número y auto incrementable, así como los campos Description y Name de tipo varchar.

MDM

	Field	Type	Extra
P	MDM_ID	NUMBER(19)	
	ENCRYTION_KEY	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null
	SFTP_PASSWORD	VARCHAR2(255 CHAR)	Allow Null
	SFTP_USR	VARCHAR2(255 CHAR)	Allow Null
	URI_PATH	VARCHAR2(255 CHAR)	Allow Null
	CATALOGUE_XSD_ID	NUMBER(19)	
	UTILITY_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100186	MDM_ID	Unique

Ilustración 4. 15 Entidad MDM Resultado T22 (Elaboración Propia).

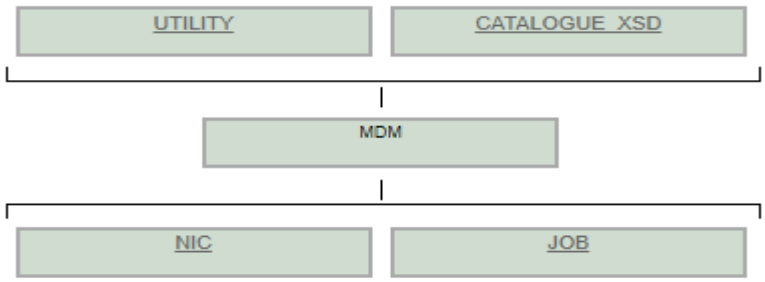


Ilustración 4. 16 Modelo de relación de MDM (Elaboración propia).

En la ilustración 4.15 se muestra la entidad MDM en esta se guarda la información del sistema MDM que se comunica con el sistema HES para enviar tareas y verificar que se realicen correctamente, cuenta con una llave primaria llamada MDM_Id de tipo número y auto incrementable, los campos Encryption_Key, Name, SFTP_Password, SFPT_User, URI_Path de tipo varchar y dos llaves foráneas llamadas Catalogue_XSD_Id de tipo número y con una relación @ManyToOne con la entidad Catalogue_XSD y Utility_Id de tipo número y con una relación @ManyToOne con la entidad Utility.

ACCESS_POINT

	Field	Type	Extra
P	ACCESS_POINT_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	FW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	FW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	HW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	HW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	IP	VARCHAR2(255 CHAR)	Allow Null
	MAC_ADDRESS	VARCHAR2(255 CHAR)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100115	ACCESS_POINT_ID	Unique

Ilustración 4. 17 Entidad Access_Point Resultado T17 (Elaboración propia).

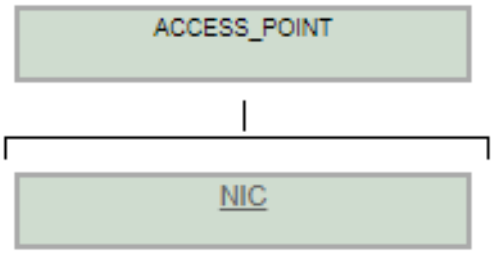


Ilustración 4. 18 Modelo De Relación de Access_Point (Elaboración propia).

En la ilustración 4.17 se muestra la entidad Access_Point esta tabla aloja la información de los Access point en campo que utiliza la Utility para conectar los dispositivos inteligentes, cuenta con una llave primaria llamada Access_Point_Id del tipo número y auto incrementable, los campos Activation_Date y Last_Modification del tipo TimeStamp, los campos FW_Revision, FW_Version, HW_Version, HW_Version, IP, Mas_Address y Seria_Number son campos con el tipo de dato Varchar.

NIC_STATUS_CATALOGUE

	Field	Type	Extra
P	NIC_STATUS_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100194	NIC_STATUS_ID	Unique

Ilustración 4. 19 Entidad Nic_Status_Catalogue Resultado T18 (Elaboración Propia).

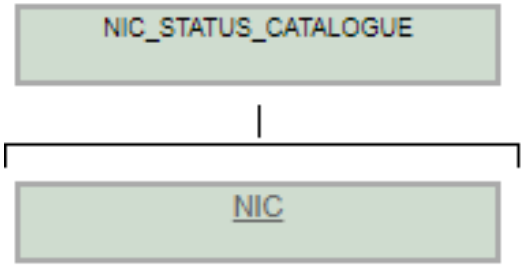


Ilustración 4. 20 Modelo de relación de Nic_Status_Catalogue (Elaboración propia).

En la ilustración 4.19 se muestra la entidad Nic_Status_Catalogue en donde se guardan los estatus en los que se puede encontrar una NIC, su nombre y una breve descripción de ella, esta entidad cuenta con una llave primaria llamada Nic_Status_Catalogue de tipo número y auto incrementable, el campo Desctiption y Name de tipo varchar.

SON

	Field	Type	Extra
P	SON_ID	NUMBER(19)	
	GATEWAY	VARCHAR2(255 CHAR)	Allow Null
	DIVISION	VARCHAR2(255 CHAR)	Allow Null
	NETWORK_ID	VARCHAR2(255 CHAR)	Allow Null
	SON_NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100196	SON_ID	Unique

Ilustración 4. 21 Entidad SON Resultado T19 (Elaboración propia).

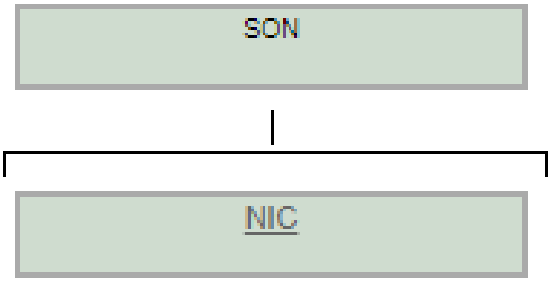


Ilustración 4. 22 Modelo de relación de SON (Elaboración propia).

En la ilustración 4.21 se muestra la entidad Son (Smart Object Network) en la cual se guarda la información de la red de los objetos formados en campo, cuenta con una llave primaria llamada Son_Id de tipo número y auto incrementable y los campos GateWay, Division, Network_Id y Son_name del tipo varchar.

NIC

	Field	Type	Extra
P	NIC_ID	NUMBER(19)	
	LAST_BILL_DATE	TIMESTAMP(6)	Allow Null
	LAST_COMUNNICATION_DATE	TIMESTAMP(6)	Allow Null
	NEXT_BILL_DATE	TIMESTAMP(6)	Allow Null
	BILLING_CYCLE	VARCHAR2(255 CHAR)	Allow Null
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	FW_NAME	VARCHAR2(255 CHAR)	Allow Null
	FW_RELEASE_DATE	VARCHAR2(255 CHAR)	Allow Null
	FW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	FW_TYPE	VARCHAR2(255 CHAR)	Allow Null
	FW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	HW_MANUFACTURER	VARCHAR2(255 CHAR)	Allow Null
	HW_MODEL	VARCHAR2(255 CHAR)	Allow Null
	HW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	HW_TYPE_DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	HW_TYPE_NAME	VARCHAR2(255 CHAR)	Allow Null
	HW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	IP_ADDRESS	VARCHAR2(255 CHAR)	Allow Null
	INSERTED_DATE	TIMESTAMP(6)	Allow Null
	LAN_MAC	VARCHAR2(255 CHAR)	Allow Null
	MODEL	VARCHAR2(255 CHAR)	Allow Null
	OPERATIONAL_STATE	VARCHAR2(255 CHAR)	Allow Null
	REMOVED	VARCHAR2(255 CHAR)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	UPDATED	VARCHAR2(255 CHAR)	Allow Null
	ACCESS_POINT_ID	NUMBER(19)	
	MDM_ID	NUMBER(19)	
	NIC_STATUS_ID	NUMBER(19)	
	SON_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100192	NIC_ID	Unique

Ilustración 4. 23 Entidad NIC resultado T21 (Elaboración propia).

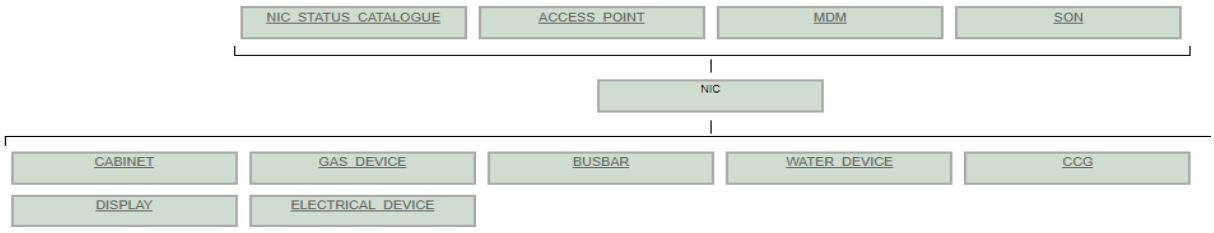


Ilustración 4. 24 Modelo de relación de NIC (Elaboración propia).

En la ilustración 4.23 se muestra la entidad NIC (Network Interface Card) es el cerebro de cada dispositivo que tiene comunicación con el GateWay, la NIC es la que se comunica con el GateWay que a su vez se comunica con el gabinete y a través de la tarjeta CCG tiene comunicación con cada uno de los medidores, cuenta con una llave primaria llamada NIC_Id de tipo número y auto incrementable, las fechas tanto de última factura, la fecha de comunicación y la próxima fecha de facturación, la fecha de inserción son de tipo TimeStamp, 22 campos de información de la tarjeta son de tipo varchar, y cuenta con 4 llaves foráneas, Access_Point, MDM_Id, Nic_Status_Id, Son_Id de tipo número.

GROUP_JOB

	Field	Type	Extra
P	GROUP_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME_JOB	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100163	GROUP_ID	Unique

Ilustración 4. 25 Entidad Group_Job Resultado T15 (Elaboración propia).

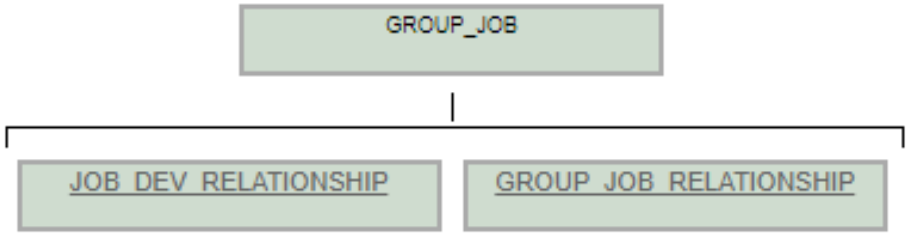


Ilustración 4. 26 Modelo de Relación De Group_Job (Elaboración propia).

En la ilustración 4.25 se muestra la entidad Group_Job en la cual se guarda la información de los grupos de tareas que serán ejecutadas, la cual cuenta con una llave primaria llamada Group_Id de tipo número, un campo con el nombre Description y name_Job de tipo Varchar.

TIME_ZONE			
	Field	Type	Extra
P	TIME_ZONE_ID	NUMBER(19)	
	VALUE	NUMBER(10)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null
	Index	Fields	Extra
	SYS_C00100203	TIME_ZONE_ID	Unique

Ilustración 4. 27 Entidad Time_Zone Resultado T10 (Elaboración Propia).

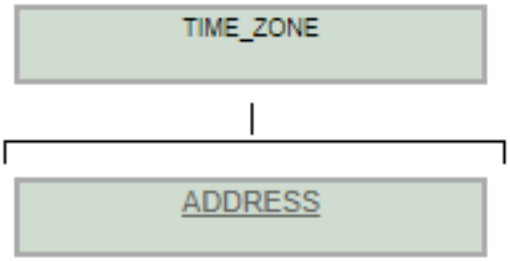


Ilustración 4. 28 Modelo de relación Time_zone (Elaboración propia).

En la ilustración 4.27 se muestra la enditad Time_Zone la cual contiene las 8 zonas en las que CFE maneja al país así como el nombre de cada una de ellas, cuenta con una llave primaria llamada Time_Zone_Id de tipo número y auto incrementable, un campo llamado value de tipo número y un campo llamado name de tipo varchar.

ADDRESS

	Field	Type	Extra
P	ADDRESS_ID	NUMBER(19)	
	CITY	VARCHAR2(255 CHAR)	Allow Null
	COUNTRY	VARCHAR2(255 CHAR)	Allow Null
	CROSS_STREET	VARCHAR2(255 CHAR)	Allow Null
	HEIGHT	VARCHAR2(255 CHAR)	Allow Null
	LATITUDE	VARCHAR2(255 CHAR)	Allow Null
	LONGITUDE	VARCHAR2(255 CHAR)	Allow Null
	STATE	VARCHAR2(255 CHAR)	Allow Null
	ZIP_CODE	VARCHAR2(255 CHAR)	Allow Null
	TIME_ZONE_ID	NUMBER(19)	Allow Null

Index	Fields	Extra
SYS_C00100117	ADDRESS_ID	Unique

Ilustración 4. 29 Entidad Address resultado T9 (Elaboración propia).

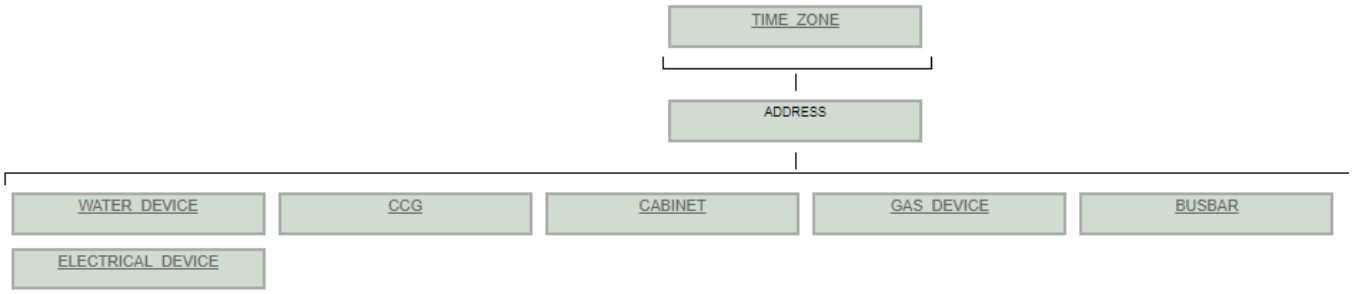


Ilustración 4. 30 Modelo de relación de Address (Elaboración propia).

En la ilustración 4.29 se muestra la entidad Address la cual contiene los datos de las direcciones las cuales son necesarias para localizar gabinetes, medidores y usuarios, cuenta con una llave primaria llamada Address_Id de tipo número y auto incrementable, los campos, city, country, Cross_Street, Heingth, latitude, Map_coordinates, State, Zip_code todos de tipo varchar, además de contar con una llave foranea llamada time_zone_Id de tipo número y con una relación @ManyToOne Con la entidad TimeZone.

DEVICE_STATUS_CATALOGUE

	Field	Type	Extra
P	DEVICE_STATUS_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100138	DEVICE_STATUS_ID	Unique

Ilustración 4. 31 Entidad Device_Status_Catalogue Resultado T12 (Elaboración propia).

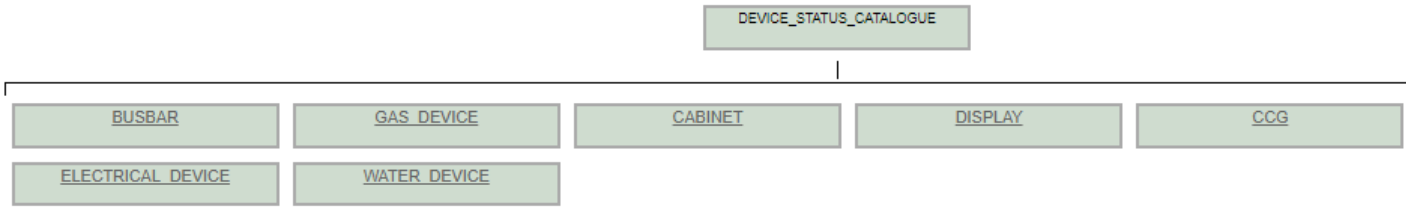


Ilustración 4. 32 Modelo de Relación de Device_Status_Catalogue (Elaboración propia).

En la ilustración 4.31 se muestra la entidad Device_Status_Catalogue esta tabla es diseñada para guardar el estatus de los medidores en campo, activo o no activo como la descripción de cada estado, cuenta con una llave primara llamada Device_Status_Catalogue_Id de tipo número y auto incrementable y los dos campos restantes son de tipo varchar.

DISPLAY

	Field	Type	Extra
P	DISPLAY_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	STATUS_LINK	VARCHAR2(255 CHAR)	Allow Null
	DEVICE_STATUS_ID	NUMBER(19)	
	NIC_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100142	DISPLAY_ID	Unique

Ilustración 4. 33 Entidad Display Resultado T13 (Elaboración propia).

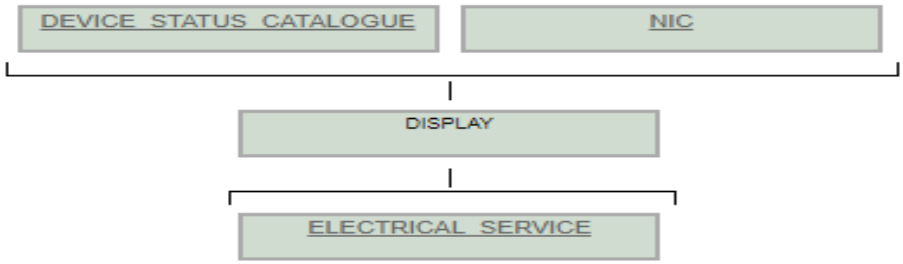


Ilustración 4. 34 Modelo de Relación de Display (Elaboración propia).

En la ilustración 4.33 se muestra la entidad Display la cual guarda información de un dispositivo llamado MRE dispositivo que consulta el consumo en tiempo real de cada usuario, cuenta con una llave primaria llamada Display_Id de tipo número y auto incrementable, cuenta con campos Activation_Date y Last_Modification de tipo TimeStamp, los campos Serial_Number, Status_Link de tipo varchar y dos llaves foráneas, Device_Status_Id de tipo number y con una relacion @ManyToOne con la entidad DeviceStatusCatalogue y Nic_Id de tipo number y con una relación @ManyToOne con la entidad NIC.

WATER_SERVICE			
	Field	Type	Extra
P	WATER_SERVICE_ID	NUMBER(19)	
	CUSTOMER_NAME	VARCHAR2(255 CHAR)	Allow Null
	Index	Fields	Extra
	SYS_C00100226	WATER_SERVICE_ID	Unique

Ilustración 4. 35 Entidad Water_Service Resultado T14 (Elaboración propia).

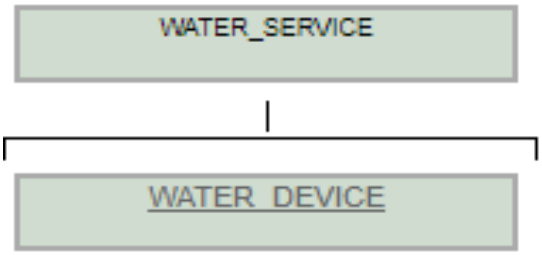


Ilustración 4. 36 Modelo de relación de Water_Service (Elaboración propia).

En la ilustración 4.35 se muestra la entidad Water_Service la cual es una tabla para escalabilidad del sistema y que pueda ser adaptado para medidores de agua, guarda la información del usuario asignado al medidor, cuenta con una llave primaria Water_Service_Id de tipo número y auto incrementable.

WATER_DEVICE

	Field	Type	Extra
P	WATER_DEVICE_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	LAST_MODIFY_ID	NUMBER(19)	Allow Null
	DEVICE_TYPE	VARCHAR2(255 CHAR)	Allow Null
	FW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	FW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	HW_MODEL	VARCHAR2(255 CHAR)	Allow Null
	HW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	HWVERSION	VARCHAR2(255 CHAR)	Allow Null
	MANUFACTURER	VARCHAR2(255 CHAR)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	ADDRESS_ID	NUMBER(19)	
	DEVICE_STATUS_ID	NUMBER(19)	
	NIC_ID	NUMBER(19)	
	WATER_SERVICE_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100224	WATER_DEVICE_ID	Unique

Ilustración 4. 37 Entidad Water_Device Resultado T11 (Elaboración propia).

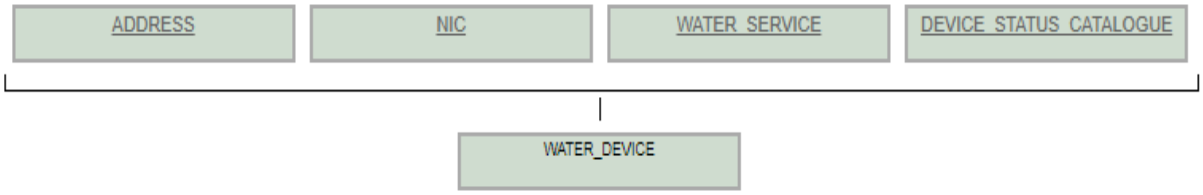


Ilustración 4. 38 Modelo de relación de Water_Device (Elaboración propia).

En la ilustración 4.37 se muestra la entidad Water_Device esta tabla fue diseñada para guardar datos de medidores inteligentes de agua, ya que el sistema HES puede ser manejado desde cualquier utility de medidores que lo requiera, cuenta con una llave primaria llamada Water_Device_Id de tipo número y auto incrementable, los campos Activation_Date, Last_Modification y Last_Modify son te tipo TimeStamp con el formato de fechas de Oracle, los campos restantes son datos del medidor y todos ellos de tipo varchar además cuenta con 4 llaves foráneas, Addres_Id de tipo número y con una relación @ManyToOne con la entidad Address, Nic_Id de tipo número y con una relación @ManyToOne con la entidad NIC, Water_Service_Id de tipo número y con una relación @ManyToOne con la entidad WaterService y Device_Status_Id de tipo número y con una relación @ManyToOne con la entidad Device_Status_Catalogue.

GAS_SERVICE			
	Field	Type	Extra
P	GAS_SERVICE_ID	NUMBER(19)	
	CUSTOMER_NAME	VARCHAR2(255 CHAR)	Allow Null
	Index	Fields	Extra
	SYS_C00100161	GAS_SERVICE_ID	Unique

Ilustración 4. 39 Entidad Gas_Service Resultado T1 (Elaboración propia).

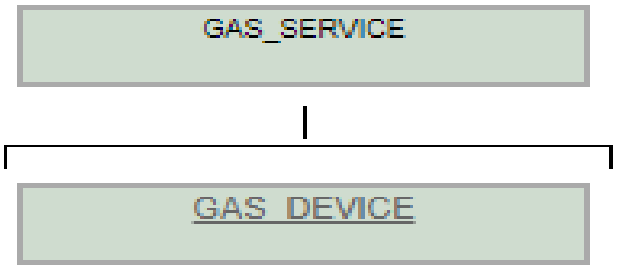


Ilustración 4. 40 Modelo de relación de Gas_Service (Elaboración propia).

En la ilustración 4.39 se muestra el diseño de la tabla Gas_Service en la cual se guarda la información relacionada a la escalabilidad del sistema pesando en ampliar el sistema para servicios de gas, la entidad cuenta con una llave primaria que es el Id identificador del tipo número y auto incrementable y un campo que guarda el nombre del cliente además de tener una relación @OneToMany con la entidad Gas_device.

GAS_DEVICE

	Field	Type	Extra
P	GAS_DEVICE_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	FW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	FW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	HW_MODEL	VARCHAR2(255 CHAR)	Allow Null
	HW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	HW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	MANUFACTURER	VARCHAR2(255 CHAR)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	ADDRESS_ID	NUMBER(19)	
	DEVICE_STATUS_ID	NUMBER(19)	
	GAS_SERVICE_ID	NUMBER(19)	
	NIC_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100159	GAS_DEVICE_ID	Unique

Ilustración 4. 41 Entidad Gas_Device Resultado de T2 (Elaboración propia).

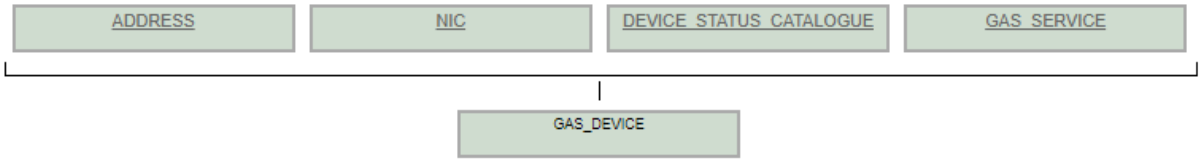


Ilustración 4. 42 Modelo de relación de Gas_Device (Elaboración propia).

En la ilustración 25 se muestra la entidad Gas_Device en la cual se guarda la información relacionada con todo el medidor inteligente de gas con un Id auto incrementable, la revisión del firmware, la versión de firmware, modelo de hardware, revisión de hardware, la revisión de hardware, información de donde fue construido, el número de serie todos ellos de tipo varchar, además cuenta con la llave foránea con una relación @OneToMany con la entidad Addres, la llave foránea Device_Status_Id con una relación @ManyToOne con la entidad DeviceStatusCatalogue, la llave foránea Gas_Service_Id con relación @ManyToOne con la entidad Gas_Service_Id y la llave foránea Nic_Id con una relación @ManyToOne con la entidad NIC.

ELECTRICAL_SERVICE

	Field	Type	Extra
P	ELECTRICAL_SERVICE_ID	NUMBER(19)	
	CUSTOMER_NAME	VARCHAR2(255 CHAR)	Allow Null
	METTER_PHASE	VARCHAR2(255 CHAR)	Allow Null
	TRANSFORMER_TYPE	VARCHAR2(255 CHAR)	Allow Null
	VOLTAGE_LEVEL_CODE	VARCHAR2(255 CHAR)	Allow Null
	DISPLAY_ID	NUMBER(19)	Allow Null

Index	Fields	Extra
SYS_C00100150	ELECTRICAL_SERVICE_ID	Unique

Ilustración 4. 43 Entidad Electrical_Service resultado de T3 (Elaboración propia).

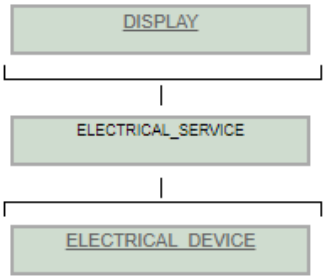


Ilustración 4. 44 Modelo de relación de Electrical_Service (Elaboración propia).

En la ilustración 4.43 se muestra la entidad Electrical_Service con una llave primaria auto incrementable que guarda la información del número de servicio que se le asigna al cliente de servicio eléctrico, cuenta con una llave primaria llamada Electrical_Service_ID con una relación @OneToMany con la entidad ElectricalService, el campo Customer_Name se guarda en nombre del cliente, el campo Metter_Phase se guarda la fase en la que se encuentra el medidor, el tipo de transformador al que es asignado de tipo varchar y el nivel de voltaje con el que cuenta de tipo varchar.

ELECTRICAL_DEVICE

	Field	Type	Extra
P	ELECT_DEV_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	HW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	BASE_TYPE	VARCHAR2(255 CHAR)	Allow Null
	CURRENT_SOCKET	VARCHAR2(255 CHAR)	Allow Null
	HW_MODEL	VARCHAR2(255 CHAR)	Allow Null
	HW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	MANUFACTURER	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null
	SW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	SW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	ADDRESS_ID	NUMBER(19)	Allow Null
	DEVICE_STATUS_ID	NUMBER(19)	
	ELECTRICAL_SERVICE_ID	NUMBER(19)	
	FORM_ID	NUMBER(19)	
	NIC_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100148	ELECT_DEV_ID	Unique

Ilustración 4. 45 Entidad Electrical_Device Resultado T04 (Elaboración propia).

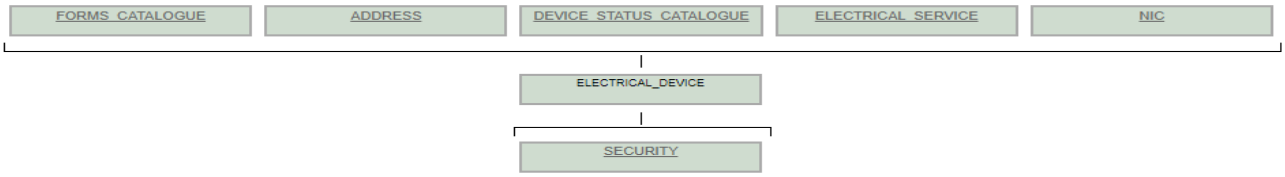


Ilustración 4. 46 Modelo de relación de Electrical_Device (Elaboración propia).

En la ilustración 4.45 se muestra la entidad Electrical_Device, cuenta con una llave primaria llamada Elect_Dev_Id la cual es auto incrementable del tipo número, donde se guarda la información de tipo de medidor, la revisión de hardware, tipo de base, el socket, modelo de hardware, la versión de hardware, donde fue construido, nombre de medidor, la versión del software, la versión del software el tipo de voltaje todos ellos del tipo varchar, cuenta con las siguientes llaves foráneas, Address_Id de tipo número y con una relación @ManyToOne con la entidad Address, Device_Status_Id de tipo número y con una relación @ManyToOne con la entidad DeviceStatusCatalogue, Display_Id de tipo número y con una relación @ManyToOne con la entidad Display, Electrical_Service_Id de tipo número y con una relación @ManyToOne con la entidad ElectricalService, NIC_Id de tipo número y con una relación @ManyToOne con la entidad NIC.

DEVICE_GROUP

	Field	Type	Extra
P	DEVICE_GROUP_ID	NUMBER(19)	
	CREATOR	VARCHAR2(255 CHAR)	Allow Null
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	GROUP_NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100134	DEVICE_GROUP_ID	Unique

Ilustración 4. 47 Entidad Device_Group Resultado T28 (Elaboración propia).

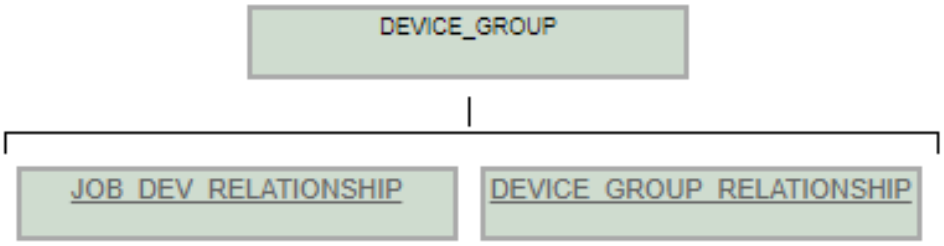


Ilustración 4. 48 Modelo de relación de Device_Group (Elaboración propia).

En la ilustración 4.47 se muestra la entidad Device_Group en la cual se guardan los grupos de medidores para que sea más fácil enviar tareas a ellos, ya sea por zona o por gabinete, esta entidad tiene una llave primaria llamada Device_Group_Id de tipo número y auto incrementable, Creator, Descriptio y Group_name de tipo varchar.

Ilustración 2- 2- 5.55 Modelo de relación de Group_Job (Elaboración propia).

TASK_TYPE

	Field	Type	Extra
P	TASK_TYPE_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100201	TASK_TYPE_ID	Unique

Ilustración 4. 49 Entidad Task_Type Resultado T27 (Elaboración propia).

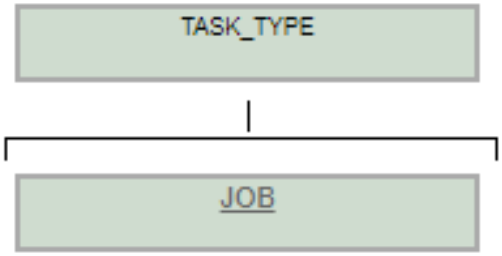


Ilustración 4. 50 Modelo de Relación de Task_Type (Elaboración propia).

En la ilustración 4.49 se muestra la entidad Task_Type en la cual se encuentra la información relacionada con el tipo de tarea a ejecutar, contiene su nombre y una breve descripción de lo que hace, la entidad cuenta con una llave primaria llamada Task_Type_Id de tipo número y auto incrementable, los campos Description y Name de tipo varchar.

CCG

	Field	Type	Extra
P	CCG_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	FW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	FW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	HW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	HW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	ADDRESS_ID	NUMBER(19)	
	DEVICE_STATUS_ID	NUMBER(19)	
	NIC_ID	NUMBER(19)	Allow Null

Index	Fields	Extra
SYS_C00100126	CCG_ID	Unique

Ilustración 4. 51 Entidad CCG Resultado T8 (Elaboración propia).

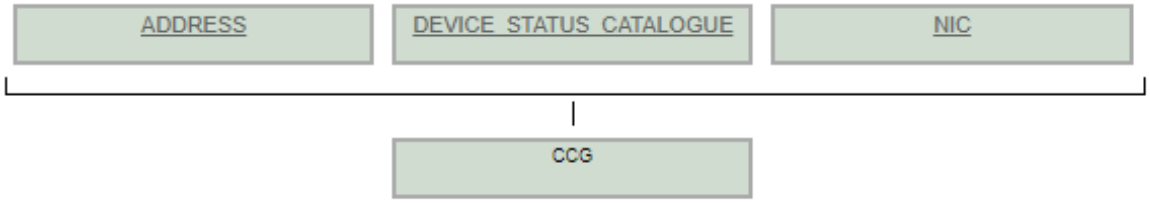


Ilustración 4. 52 Modelo de relación CCG (Elaboración Propia).

En la ilustración 4.51 se muestra la entidad CCG llamada así por su acrónimo Centro De Control de Gabinete (información no disponible por la empresa), la cual contiene una llave primaria llamada CCG_Id de tipo número, la entidad contiene datos de la tarjeta todos ellos de tipo varchar y cuenta con una llave foranea llamada Address_Id de tipo número con una relación @ManyToOne con la entidad Address.

BUSBAR

	Field	Type	Extra
P	BUSBAR_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	FW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	FW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	HW_REVISION	VARCHAR2(255 CHAR)	Allow Null
	HW_VERSION	VARCHAR2(255 CHAR)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	ADDRESS_ID	NUMBER(19)	
	DEVICE_STATUS_ID	NUMBER(19)	
	NIC_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100122	BUSBAR_ID	Unique

Ilustración 4. 53 Entidad BusBar resultado T06 (Elaboración propia).

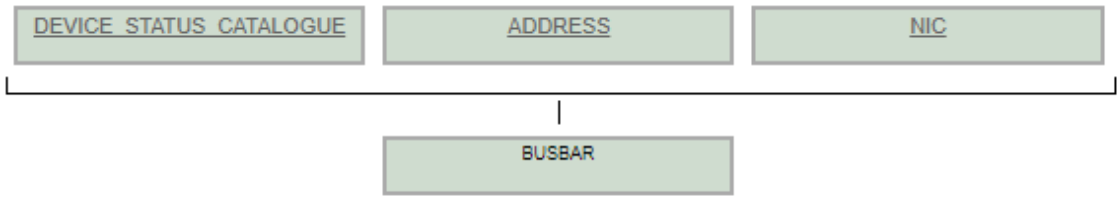


Ilustración 4. 54 Modelo de Relación de BusBar (Elaboración propia).

En la ilustración 4.53 se muestra la entidad BusBar (Por política de la empresa no se puede describir) la cual contiene la información de la tarjeta, revisión de firmware, versión de firmware, revisión de hardware y el número de serie son de tipo varchar, además cuenta con una llave foranea llamada Addres_Id de tipo número y que tiene una relación @ManyToOne con la entidad Address.

SECURITY

	Field	Type	Extra
P	SECURITY_ID	NUMBER(19)	
	CUSTOMER_PASSWORD	VARCHAR2(255 CHAR)	Allow Null
	BILLING_PASSDWORD	VARCHAR2(255 CHAR)	Allow Null
	NIVEL	VARCHAR2(255 CHAR)	Allow Null
	UNREGISTRED_PASSWORD	VARCHAR2(255 CHAR)	Allow Null
	ELECT_DEV_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100199	SECURITY_ID	Unique

Ilustración 4. 55 Entidad Security Resultado T05 (Elaboración propia).

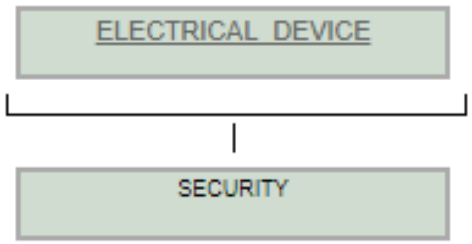


Ilustración 4. 56 Modelo de relación de Security (Elaboración propia)

En la ilustración 4.55 se muestra la entidad Security, cuenta con una llave primaria llamada Secu_Id auto incrementable, esta tabla contiene la información de seguridad de los usuarios, el cusstomer_password, nivel, unregistred_password de tipo varchar y cuenta con una llave foránea llamada Electr_Dev_Id con relación @ManyToOne con la entidad ElectricalDevice.

CABINET			
	Field	Type	Extra
P	CABINET_ID	NUMBER(19)	
	ACTIVATION_DATE	TIMESTAMP(6)	Allow Null
	LAST_MODIFICATION	TIMESTAMP(6)	Allow Null
	SERIAL_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	ADDRESS_ID	NUMBER(19)	Allow Null
	DEVICE_STATUS_ID	NUMBER(19)	
	NIC_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100130	CABINET_ID	Unique

Ilustración 4. 57 Entidad Cabinet Resultado T07 (Elaboración propia).

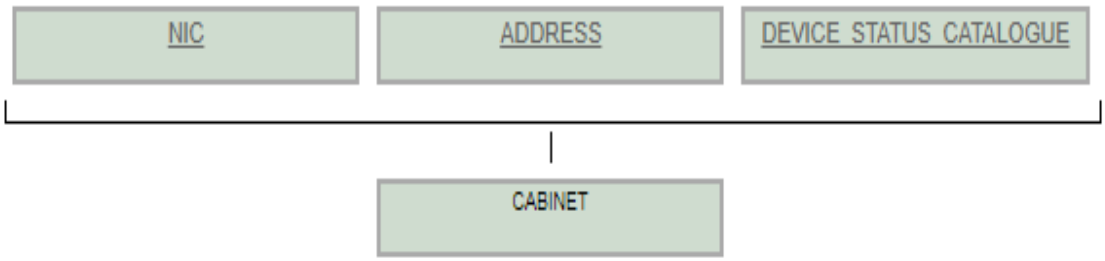


Ilustración 4. 58 Modelo de relación Cabinet (Elaboración propia).

En la ilustración 4.57 se muestra la entidad Cabinet en la cual se guarda la información del gabinete donde son asignados los medidores en campo, cuenta con una llave primaria llamada Cabinet_Id de tipo número y un campo de número de serie de tipo Varchar con una llave foranea llamada Address_Id con relación @ManyToOne con la entidad Address.

DEVICE_GROUP_RELATIONSHIP

	Field	Type	Extra
P	DEVICE_GROUP_RELATIONSHIP_ID	NUMBER(19)	
	DEVICE_ID	NUMBER(19)	Allow Null
	DEVICE_GROUP_ID	NUMBER(19)	Allow Null

Index	Fields	Extra
SYS_C00100136	DEVICE_GROUP_RELATIONSHIP_ID	Unique

Ilustración 4. 59 Entidad Device_Group_Relationship Resultado T29 (Elaboración propia).

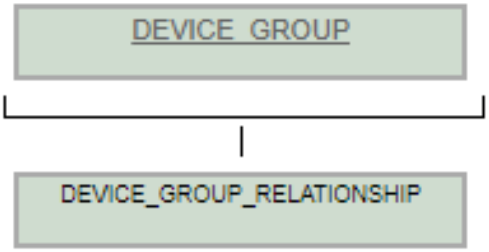


Ilustración 4. 60 Modelo de Relación Device_Group_Relationship (Elaboración propia).

En la ilustración 4.59 se muestra la entidad Device_Group_Relationship en esta tabla se guarda la información de las relaciones que se va a tener entre los grupos de dispositivos y es creada para la utilización de lógica de negocios de la aplicación HES, cuenta con una llave primaria llamada Device_Group_Relationship de tipo número y auto incrementable además un campo Llamado Device_Id de tipo número y una llave foránea llamada Device_Group_Id de tipo número y con una relación @ManyToOne con la entidad Device_Group.

JOB_STATUS_CATALOGUE

	Field	Type	Extra
P	JOB_STATUS_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null

Index	Fields	Extra
SYS_C00100179	JOB_STATUS_ID	Unique

Ilustración 4. 61 Entidad Job_Status_Catalogue Resultado T25 (Elaboración propia).

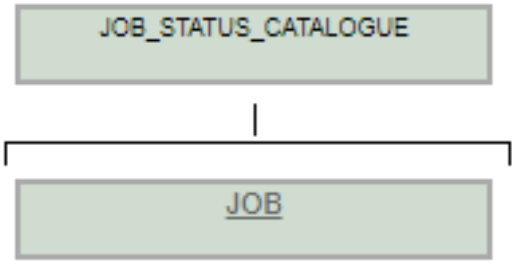


Ilustración 4. 62 Modelo de relación Job_Status_Catalogue (Elaboración propia).

En la ilustración 4.61 se muestra la entidad Job_Status_Catalogue, esta tabla se crea para guardar los estatus en lo que se puede encontrar un JOB, Ejecutada, En espera y no ejecutada cuenta con una llave primaria llamada Job_Status_Id de tipo número y auto incrementable, los campos Description y Name de tipo varchar.

JOB

	Field	Type	Extra
P	JOB_ID	NUMBER(19)	
	CREATION_DATE	TIMESTAMP(6)	Allow Null
	DEPLOY_DATE	TIMESTAMP(6)	Allow Null
	END_DATE	TIMESTAMP(6)	Allow Null
	LAST_RUN_STATUS_TIME	TIMESTAMP(6)	Allow Null
	INTERVAL	NUMBER(10)	Allow Null
	CREATOR_ID	NUMBER(19)	Allow Null
	DST	VARCHAR2(255 CHAR)	Allow Null
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	ACTIVATION_STATUS	VARCHAR2(255 CHAR)	Allow Null
	LAST_EXECUTION_STATUS	VARCHAR2(255 CHAR)	Allow Null
	ONFAILURE	VARCHAR2(255 CHAR)	Allow Null
	OPERATION_NUMBER	VARCHAR2(255 CHAR)	Allow Null
	PRIORITY	VARCHAR2(255 CHAR)	Allow Null
	REPETITION	VARCHAR2(255 CHAR)	Allow Null
	RETRIES	VARCHAR2(255 CHAR)	Allow Null
	TYPE	VARCHAR2(255 CHAR)	Allow Null
	JOB_STATUS_ID	NUMBER(19)	
	MDM_ID	NUMBER(19)	
	TASK_TYPE_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100172	JOB_ID	Unique

Ilustración 4. 63 Entidad JOB resultado T26 (Elaboración propia).

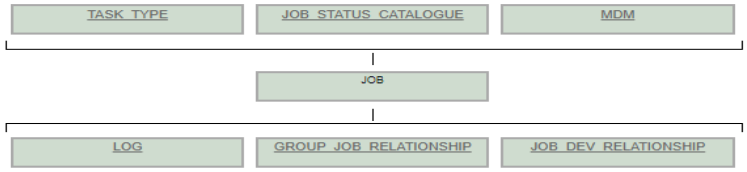


Ilustración 4. 64 Modelo de relación de JOB (Elaboración propia).

En la ilustración 4.63 se muestra la entidad JOB en esta tabla se guarda información de las Jobs (Nombre dado a las tareas) incluye desde la fecha de creación, la fecha de implementación, la fecha de terminación, quien crea la tarea, la prioridad de la tarea, cuantas veces se intentó y su tipo, cuenta con una llave primaria llamada JOB_Id de tipo número y es auto incrementable, los campos Creation_Date, Deploy_Date, End_Date, Last_Run_Status_Time de tipo DateStamp, los campos Interval y Creator_Id de tipo número, los campos restantes de tipo varchar, asi como 3 llaves foráneas, Job_Status_Id de tipo número y con una relación @ManyToOne con la entidad JobStatusCatalogue, MDM_Id de tipo número y con una relación @ManyToOne con la entidad MDM y Task_Type_Id de tipo número y con una relación @ManyToOne con la entidad Task_Type.

LOG

	Field	Type	Extra
P	LOG_ID	NUMBER(19)	
	OPERATION_RESULT	NUMBER(1)	Allow Null
	EXECUTION_DATESTAMP	TIMESTAMP(6)	Allow Null
	RETRY_COUNT	NUMBER(10)	Allow Null
	JOB_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100182	LOG_ID	Unique

Ilustración 4. 65 Entidad LOG Resultado T24 (Elaboración propia).

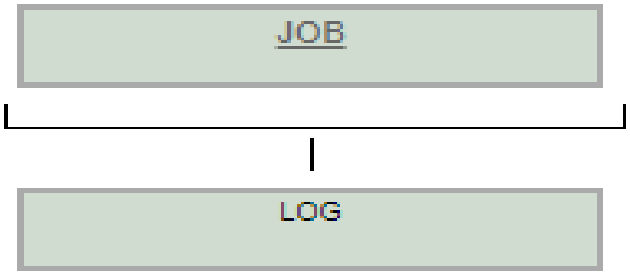


Ilustración 4. 66 Modelo de Relación de LOG (Elaboración propia).

En la ilustración 4.65 se muestra la entidad LOG, esta tabla se crea para guardar la información de historial en el sistema HES, aloja si las operaciones fueron correctas o si la tarea fracasa además de la fecha en la que la tarea fue ejecutada y la veces que fue re intentada, contiene una llave primaria llamada LOG_Id de tipo número y auto incrementable, los campos Operation_Result, Execution_Date, Retry_Count de tipo varchar y una llave foránea llamada Job_Id de tipo número y con una relación @ManyToOne con la entidad JOB.

GROUP_JOB_RELATIONSHIP

	Field	Type	Extra
P	GROUP_JOB_RELATIONSHIP_ID	NUMBER(19)	
	DESCRIPTION	VARCHAR2(255 CHAR)	Allow Null
	NAME	VARCHAR2(255 CHAR)	Allow Null
	GROUP_ID	NUMBER(19)	
	JOB_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100167	GROUP_JOB_RELATIONSHIP_ID	Unique

Ilustración 4. 67 Entidad Group_Job_Relationship Resultado T31 (Elaboración propia).

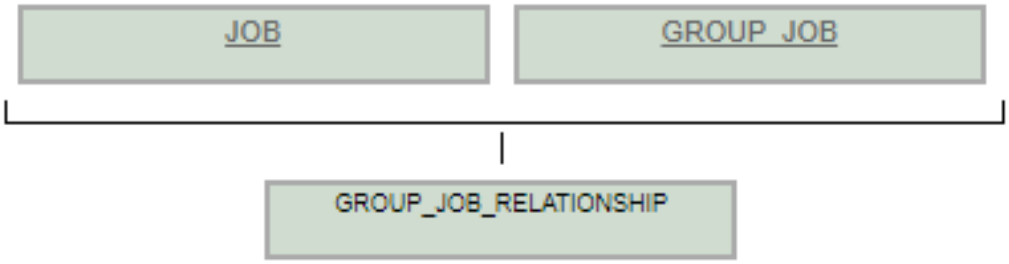


Ilustración 4. 68 Modelo de relación de Group_Job_Relationship (Elaboración propia).

En la ilustración 4.67 se muestra la entidad Group_Job_Relationship la cual guarda la información del grupo de tareas para relacionar un grupo con una tarea la cual tiene el nombre del grupo y una breve descripción del grupo y es utilizado por la lógica de negocios del sistema, cuenta con una llave primaria Group_Job_Relationship_Id de tipo número y auto incrementable, los campos Name y Description de tipo varchar, cuenta con dos llaves foráneas, Group_Id de tipo número y con una relación @ManyToOne con la entidad Group_Job y Job_Id de tipo número y con una relación @ManyToOne con la entidad Job

JOB_DEV_RELATIONSHIP

	Field	Type	Extra
P	JOB_DEV_REL_ID	NUMBER(19)	
	DEVICE_ID	NUMBER(19)	Allow Null
	DEVICE_TYPE	VARCHAR2(255 CHAR)	Allow Null
	DEVICE_GROUP_ID	NUMBER(19)	
	GROUP_ID	NUMBER(19)	
	JOB_ID	NUMBER(19)	

Index	Fields	Extra
SYS_C00100177	JOB_DEV_REL_ID	Unique

Ilustración 4. 69 Entidad Job_Dev_Relationship Resultado T30 (Elaboración propia).

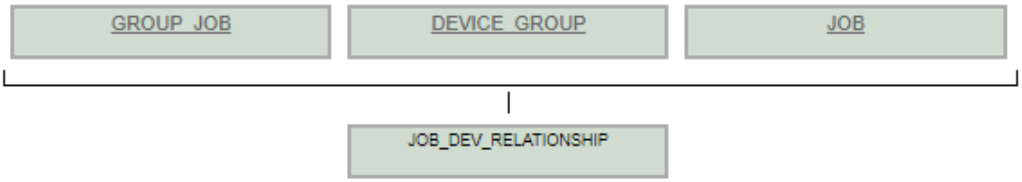


Ilustración 4. 70 Modelo de relación de Job_Dev_Relationship (Elaboración propia).

En la ilustración 4.69 se muestra la entidad Job_Dev_Relationship esta tabla se crea para el uso de la lógica de negocios de la aplicación se utiliza para las relaciones que tienen los dispositivos con las tareas, cuenta con una llave primaria llamada Job_Dev_Rel_Id de tipo número y auto incrementable, cuenta con un campo llamada Device_Id de tipo número y un campo llamado Device_Type de tipo varchar, además de tres llaves foráneas, Device_Group_Id de tipo número y con una relación @ManyToOne con la entidad Device_Group, Group_Id de tipo número con una relación @ManyToOne con la entidad Group_Job y Job_Id de tipo número y con una relación @ManyToOne con la entidad JOB.

4.4. - Resultados del cuarto modulo llamado “Spring Security”.

Para el desarrollo del cuarto modulo entregable del sistema HES el modulo se debía adaptar con la vista debido a problemas con las dependecias con las que se desarrollaron era un problema fucionar dependencias con Spring Security, el modulo quedo desarrollado listo para adaptar con sistemas que necesiten seguridad web con roles y usuarios, por lo que se quedo como trabajo a futuro el fucionar el sistema HES con Spring Security, los resultados obtenidos por las tareas de los Sprints (Tareas) fueron las siguientes;

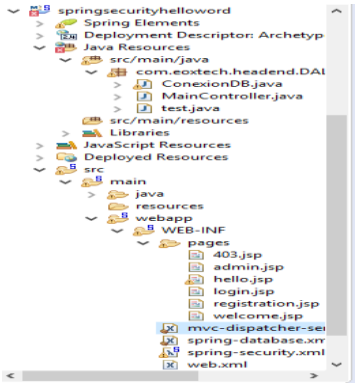


Ilustración 4. 71 Resultado de T1 del cuarto modulo. (Elaboracion propia).

En la ilustración 4.71 se muestra el resultado de la tarea 1 donde se observa el espacio de trabajo del modulo de spring security.

```
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-web</artifactId>
	<version>${spring-security.version}</version>
</dependency>
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-config</artifactId>
	<version>${spring-security.version}</version>
</dependency>
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-taglibs</artifactId>
	<version>${spring-security.version}</version>
</dependency>
```

Ilustración 4. 72 Resultado de T02 del cuarto modulo (Elaboración propia).

En la ilustración 4.72 se muestran las dependencias agreagdas al archvio POM.xml donde se descargaran los archivos .jar de las que depende Spring Security

Para la tarea 6 se configura el archvio web.xml donde se confirugura el mvc-dispatcher y el Servlet a utilizar, se integra el listener Dispatcher y se cargan los archivos de spring security, tanto Spring-security.xml como spring-database.xml.

En la tarea 4 se configura archvio spring-security.xml donde se muestra la configuración para los roles de usuario tanto el rol de Admin como el root, además se configura la consulta que se utilza para los roles de usuario.

En la tarea 5 se configura el archivo Sprin-database.xml donde se crea la coneccion con Oracle para inicio de sesión de Spring.

En la tarea 6 se hace la configuración del archivo de mapeo de las vistas que serán escuchadas por el Dispatcher.

USERS

	Field	Type	Extra
P	USERNAME	VARCHAR2(36 CHAR)	
	EMAIL	VARCHAR2(255 CHAR)	
	ENABLED	NUMBER(1)	
	FULL_NAME	VARCHAR2(255 CHAR)	
	PASSWORD	VARCHAR2(255 CHAR)	

Index	Fields	Extra
SYS_C0092935	USERNAME	Unique

Ilustración 4. 73 Resultado del modelo E-R Entidad Users (Elaboración propia).

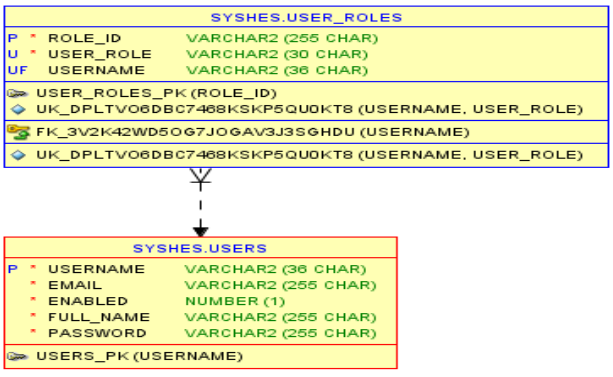


Ilustración 4. 74 Modelo de relación de la entidad Users (Elaboración propia).

En la ilustración 4.73 se muestra el resultado de la creación de la entidad Users del modelo E-R y es donde se guardan los datos de los usuarios que tendrán acceso al sistema.

USER_ROLES

	Field	Type	Extra
P	ROLE_ID	VARCHAR2(255 CHAR)	
	USER_ROLE	VARCHAR2(30 CHAR)	
	USERNAME	VARCHAR2(36 CHAR)	Allow Null

Index	Fields	Extra
SYS_C0092929	ROLE_ID	Unique

Ilustración 4. 75 Resultado del modelo E-R de Spring security Entidad User_Roles (Elaboración propia).

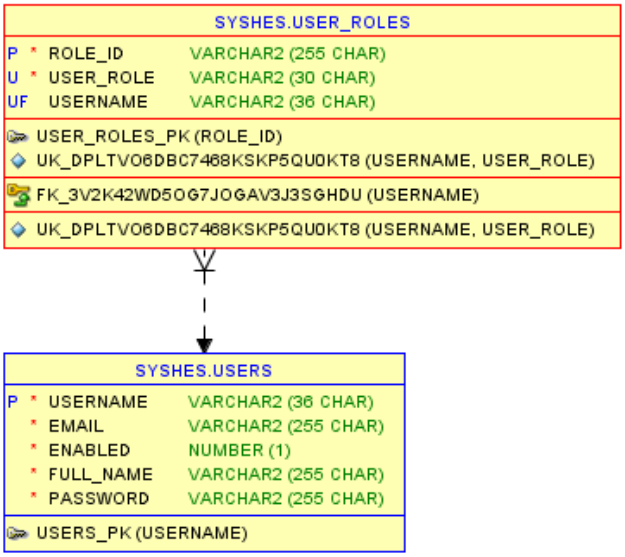


Ilustración 4. 76 Modelo de relación de la entidad User-roles (Elaboración propia).

En la ilustración 4.75 se muestra la entidad Roles del modelo E-R la cual tiene relacion @ManyToOne con la entidad Users, y en la cual se guardan los roles que se asignan para dar acceso a partes del sistema que sean definidos.

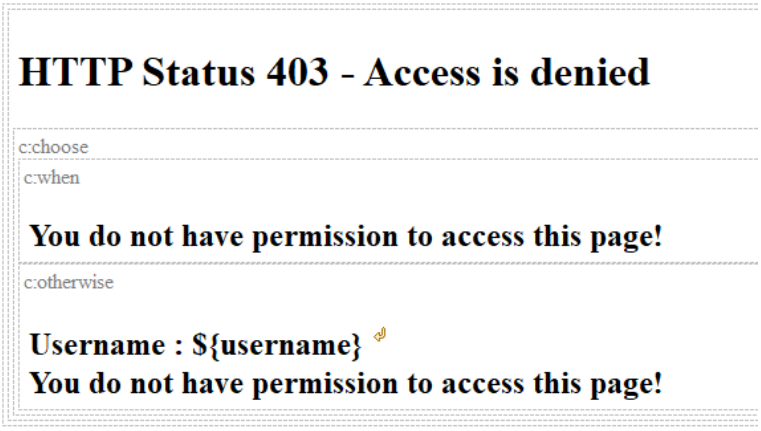


Ilustración 4. 77 Resultado de la tarea 7 del cuarto modulo entregable (Elaboración propia)

En la ilustración 4.77 se muestra la pagina 403.jsp, la cual manda un mensaje de error, para las secciones que puede ver el usuario logeado, la primera parte el acceso solo es para usuarios con credencial de administrador, la segunda sección es para usuarios Root.

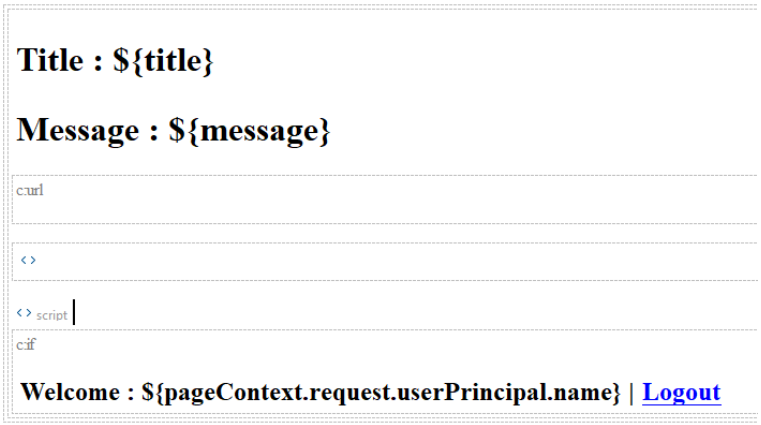


Ilustración 4. 78 Resultado de la tarea 8 del la historia 4 (Elaboración propia).

En la ilustración 4.78 se muestra la pagina Admin.jsp la cual controla lo que el usuario puede y no puede ver dentro del sistema.

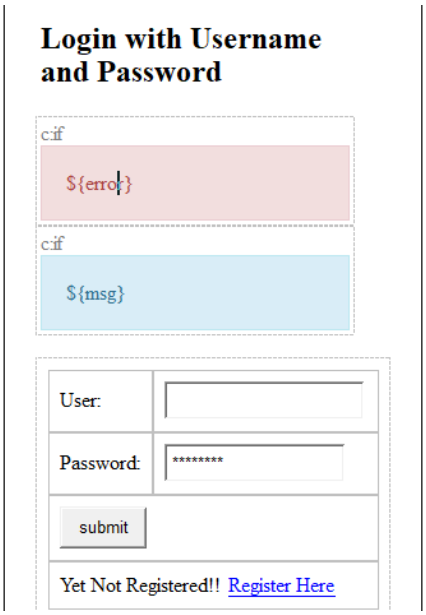


Ilustración 4. 79 Resultado de la tarea 9 de la historia 4 (Elaboración propia).

En la ilustración 4.79 se muestra la pagina Login.jsp la cual contiene el mensaje de error al no contar con credenciales para el sistema, y los campos requeridos para que el usuario entre al sistema.

4.5. - Resultados.

Para el sistema HES el ingeniero Manuel Flores Nava, realizo pruebas a los modulos para que fueran completamente funcionales, los resultados de dichas puebas se aprecian en (Nava, 2017), apreciados en las ilustraciones

RESULTADOS DE LA BASE DE DATOS				
Requerimientos Tecnicos	**Casos de prueba**	**TIPO DE PRUEBAS**	**RESULTADOS**	**Observaciones**
Conexión a la base de datos	Acceso a la base de datos	CAJA NEGRA (Entorno cliente-servidor)	SI	
Consulta de nombre tablas	Nombre de tablas y campos correspondientes	Consultas Análisis estático	No	Se notifica a desarrollador
Nivel de acceso a la base de datos	Acceso como supe usuario	Caja negra Inyección de código	SI	
	Acceso como administrador	Caja negra Inyección de código	SI	
	Acceso como consulta	Caja negra Inyección de código	No	Se notifica a desarrollador
	Implementación de *Spring security*	Caja negra Inyección de código	No	Se notifica a desarrollador

Ilustración 4. 80 Resultados al modulo de la base de datos Adaptada de (Nava, 2017).

En la ilustración 4.80 se muestran los resultados de las pruebas a base de datos realizadas, acceso a base de datos, nombre de campos y tablas correspondientes, acceso como super usuario, acceso como administrador, acceso con consulta, implementación de spring.

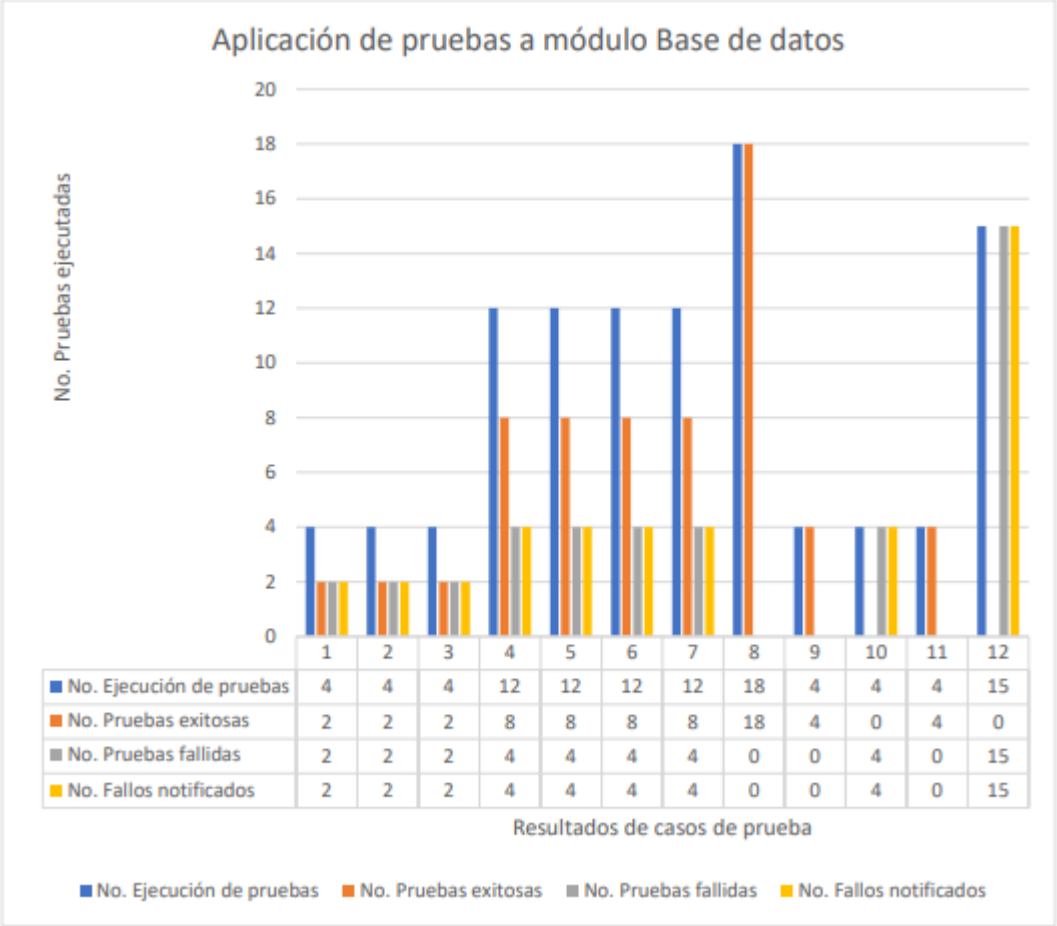


Ilustración 4. 81 Aplicacion de pruebas al modulo base de datos adaptado de (Nava, 2017).

En la ilustación 4.81 se muestran los resultados finales de las pruebas hechas a la base de datos relacional.

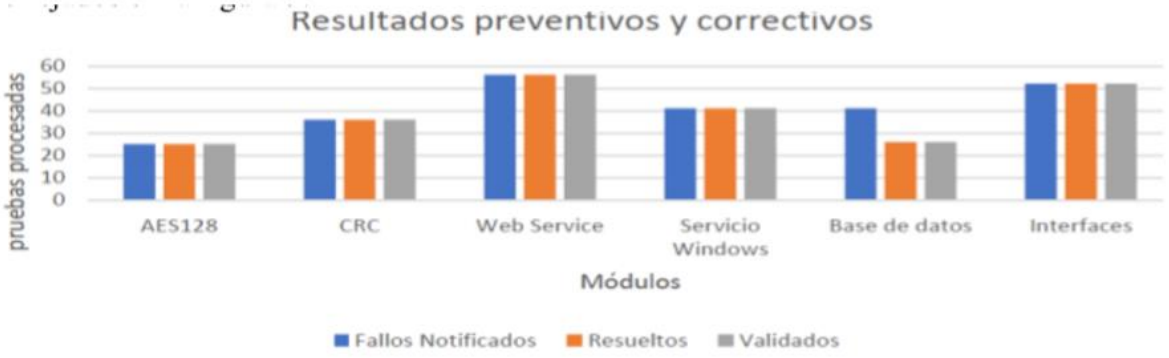


Ilustración 4. 82 Resultados preventivos y correctivos adaptado de (Nava, 2017).

La ilustración 4.82 muestra los resultados finales al sistema Head End System.

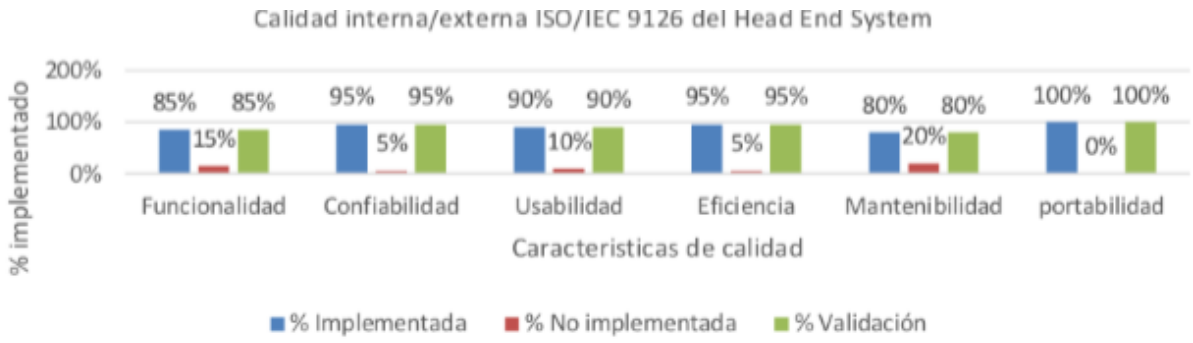


Ilustración 4. 83 Resultados de calidad hechas al sistema HES. Adaptado de (Nava, 2017).

Capítulo 5. Conclusiones y trabajos futuros.

El desarrollo del proyecto descrito en la presente tesis cumplió con las expectativas de desempeño del producto final entregado a la empresa EOSTech, quedando como registro la carta de satisfacción del cliente incluida en los anexos de este documento.

La implementación de metodologías agiles para el desarrollo de este módulo mencionadas en el capítulo 3 brindo grandes ventajas en cuanto a la rapidez de programación y en entendimiento de la funcionalidad esperada.

La asignación de tareas por parte del jefe de proyecto fue basada en las habilidades de cada uno de los integrantes, una vez establecidos los módulos que conforman el Head End System se tuvo que considerar la interacción con los demás módulos.

El módulo de URL única (Sección 3.3) es el encargado de brindar seguridad a las URL del sistema, para evitar inyecciones de código a la base de datos y accesos no deseados a la información.

El módulo de diseño de base de datos (sección 3.4) fue la conclusión obtenida por estudios basado en precios, seguridad, espacios de almacenamiento y facilidades de manejo, por lo que el mejor candidato esto fue Oracle, un gestor robusto, poderoso y confiable con grandes capacidades de almacenamiento, además se eligió una base de datos relacional por el tipo de sistema que es, ya que el sistema HES requiere información de diferentes tablas para mostrar la información que es requerida por la interfaz gráfica además de que obliga al cumplimiento de condiciones acerca de la base de datos, además de una gramn ventaja para evitar la diplicidad de registros garantiza la integridad referencial, asi al eliminar un registro elimina todos los registros relacionados independientes.

Para el modelado de la base de datos dentro del sistema se utilizó el Framework de JPA e Hibernate el cual se mencionó en la sección 2.14 de este documento se encarga de modelar los objetos de datos y java Beans de cada una de las tablas para facilitar el manejo de la información

a los desarrollares de lógica de negocios además de hacer mas rápida la obtención de datos gracias a la persistencia de ellos.

El modelado de la base de datos descrito en la sección 3.4 surgio a partir de las entidades pertenecientes al sistema HES por lo que estos hacen más fácil los procesos de comisión federal de electricidad, gracias a esto se seleccionaron los campos indispensables con los que debía contar, con el paso del tiempo la base de datos fue creciendo y cambiando agregando nuevas tablas y campos según lo requerido, el resultado descrito durante el capítulo 4 de este documento presenta una base de datos contenedora de información referente a los usuarios, a dispositivos inteligentes, las Utility que necesiten la base. Referente a la escalabilidad del sistema y pensando no solo en ramo eléctrico el sistema es adaptable a cualquier tipo de base de datos que se requiera (MySQL, H2 y Oracle) en el archivo de configuración Hibernate cambiando los dialectos a las librerías que se necesiten.

Una vez modelada la base de datos se comenzó con las relaciones de cada tabla descritas en la sección 3.4 estableciendo cada una de ellas conforme la lógica de negocios lo requería.

Para el cuarto modulo del sistema HES en la sección 3.5 de este documento se desarrolló el sistema de seguridad con roles y usuarios, pero debido a incompatibilidad de dependencias el modulo no se pudo integrar al sistema, por lo que queda como trabajo a futuro.

Por último se puede concluir que se ha comprobado que es posible el desarrollo de una base de datos relacional utilizando ORM para hacer más eficientes y robustas las transacciones de un HEAD END SYSTEM. Con metodologías de desarrollo ágil, con esto se contesta la pregunta de investigación en la sección 1.5 de este documento.

5.1. - Trabajos a Futuro.

Como trabajos a futuro se establecieron las pruebas con un servidor web más potente, apache TomCat es demasiado pequeño para el tamaño del sistema, por lo que se migrara a Tomee que es una versión Enterprise de apache TomCat.

Para la seguridad del sistema se estableció que los usuarios logeados al sistema cuenten con un límite de tiempo de 30 minutos por cada sesión que tengan.

Además de puntos para robustecer la base de datos.

- Robustecer la seguridad de inicio de Sesión en Oracle como súper usuario.
- Integración del módulo Spring Security, no solo el prototipo.

Glosario de términos.

Ajax.- acrónimo de Asynchronous JavaScript And XML (JavaScript asíncronoy XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications).

API.- API son las siglas de Interfaz de Programación de Aplicaciones, en español. Concretamente, es una especificación formal sobre cómo un módulo de un software se comunica o interactúa con otro

Atributo.- Los atributos son las características individuales que diferencian un objeto de otro y determinan su apariencia, estado u otras cualidades. Los atributos se guardan en variables denominadas de instancia, y cada objeto particular puede tener valores distintos para estas variables.

Clase.- Es una construcción que permite crear tipos personalizados propios mediante la agrupación de variables de otros tipos, métodos y eventos. Una clase es como un plano. Define los datos y el comportamiento de un tipo.

Comunicación multipunto.- Es cuando dos o más líneas de conexión o terminales comparten porciones de una línea común.

Dirección IP.- Las direcciones IP (IP es un acrónimo para Internet Protocol) son un número único e irrepetible con el cual se identifica una computadora conectada a una red que corre el protocolo IP.

Dirección Mac.- La dirección MAC es un identificador único que cada fabricante le asigna a la tarjeta de red de sus dispositivos conectados, desde un ordenador o móvil hasta routers, impresoras u otros dispositivos como tu Chromecast. Sus siglas vienen del inglés, y significan Media Access Control.

Diseño web.- El diseño web es una actividad que consiste en la planificación, diseño e implementación de sitios web

Entidad.- Representación de un objeto de la vida real en bases de datos.

Framework.- Desde el punto de vista del desarrollo de software, un framework es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado.

Hibernate.- Es una herramienta de Mapeo objeto-relacional para la plataforma Java (y disponible también para .Net con el nombre de Hibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

HQL.- Hibernate ofrece también un lenguaje de consulta de datos llamado HQL (Hibernate Query Language) similar a SQL.

IDE.- (Integrated Development Environment – Entorno integrado de desarrollo). Aplicación compuesta por un conjunto de herramientas útiles para un programador.

Instanciado.- El término instancia se refiere siempre a un objeto creado por el objeto-clase en tiempo de ejecución y por supuesto, pueden tener propiedades y métodos.

Interfaz.- En software, parte de un programa que permite el flujo de información entre un usuario y la aplicación, o entre la aplicación y otros programas o periféricos

Mapeo.- Es un archivo XML que describe las características de los POJOS.

Persistencia.- Se llama “persistencia” de los objetos a su capacidad para guardarse y recuperarse desde un medio de almacenamiento.

POJOS (Plain Old Java Object).- Es una implementación de un bean el cual da persistencia a los datos obtenidos desde la base de datos.

Referencias

Alegsa. (2009). *Departamento de Tratamiento de la Información*. Retrieved from http://www.iec.csic.es/criptonomicon/java/quesjava.html

Babu, C. (2016). DESH: Database evaluation system with hibernate ORM framework. *Communications and Informatics (ICACCI).*

Bak, T., Sakowicz, B., & Napieralski, A. (2006). Development Of Advanced J2EE Solutions Based On Lightweight Containers On The Example Of "e-department" Application. *Proceedings of the International Conference Mixed Design of Integrated Circuits and System, 2006. MIXDES 2006.*

Cargnelutti, F. (2005, Febrero). *Persistencia de Objetos Java utilizando Hibernate,*. Retrieved from http://www.programacion.com/articulo/persistencia_de_objetos_java_utilizando

Caules, C. Á. (2009). *Arquitectura Java JPA.*

CFE. (2007, Abril 19). Retrieved from https://app.cfe.mx/informe2007/08_m.html

Crespo. (2007). *Introducción a Hibernate*. Retrieved from http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=hibernate

Devi, M. N., & Priya, A. (2016). Invoicing and analytics for small and micro manufacturing enterprises. *Conference on Recent Trends in Information Technology (ICRTIT).*

Farhangi, H. (2012). *The path of the smtar grid.* Us: IEEE.

Foundation, A. S. (2018). *Apache Tomcat*. Retrieved from http://tomcat.apache.org/

Galli, P. (2005, 11 2). *Eclipse Public License*. Retrieved from http://www.eclipseplugincentral.com/

Gómez, R. (2015, Noviembre 11). *Modelo Vista Controlador*. Retrieved from http://rodrigogr.com/blog/modelo-vista-controlador/

Grids, E. S. (2012, 02 12). *Smart Grids European Technology Platform*. Retrieved from http://smartgrids.eu

Gungor Vehbi, L. B. (2010). *Opportinities and changes of wireless sensor.*

Harris, I. (2012). Differences Between Controller Continnium. *Frescale Application*, 1-16.

Higa, S. (2013). *Freescale semiconductor application.* ES: FFT-Based.

Himani Pandey, H. R. (2018). Web-based network management system implemented using Hibernate, JBoss and spring framework. *2017 2nd International Conference on Telecommunication and Networks (TEL-NET).*

IEEE. (2012, febrero 15). *Smart Grid Wahts is a Smart Grid*. Retrieved from http://smartgrid.ieee.org

IEEE, I. S. (2012, febrero). *IEEE Smart Grid*. Retrieved from http://smartgrid.ieee.org/ieee-smart-grid

Inga, H. (2012). *Calibration And Procedure And Pogramable.* ES: FreeScale.

Institute, E. P. (2009). *Smart Grid Demostrations Overview.*

Institute, E. P. (2010). *Accuaracy of digital Electricity Meters.* California.

Kendall, K. E. (2007). *Introducción al diseño de base de datos.*

Kes, T. (2012). A Low-Cost Single-Phase electricity Meter. In T. Kes, *A Low-Cost Single-Phase electricity Meter* (pp. 1-19). US: Texas Instrument.

Knirsch, P. (2012). Performance Assessment whit Algoritms used in metering applications. In P. Knirsch, *Performance Assessment whit Algoritms used in metering applications* (pp. 1-32). US.

Kroenke, D. (2013). *Procesamiento de Base de Datos.* México: Pearson.

Laboratory, N. E. (2009). *Modern Grid Strategy Powering.* NETL.

Language, T. S. (2015). Pu Xiaowei. *Intelligent Computation Technology and Automation (ICICTA).*

Luis Puebla, A. G. (2013). *LSD Driver Spesification.* Mexico.

Mellon, C. (2012, 02 16). *Smart Grid Research*. Retrieved from http://www.cmu.edu/homepage/enviroment/2010/summer/smart-grid-research.shtml

Mera, B. (2011, Mayo 12). *Designin Smart Meters for the mstar grid*. Retrieved from http://www.freescale.com/files/training.pdf

Meurice, L., Nagy, C., & Cleve, A. (2016). Detecting and Preventing Program Inconsistencies under Database Schema Evolution. *2016 IEEE International Conference on Software Quality, Reliability and Security (QRS)*.

MQX-Enabled. (2012). *Single-Phase Electricity.* ES: MCF51.

mrd. (n.d.).

Nava, M. F. (2017). Diseño de una metodologia de testing en scrum para un sistema integral de tipo Head End System enfocado a Smartcities. *Academia Journals*.

NIST, S. G. (2010, Noviembre 15). *NIST and the Smart Grig*. Retrieved from http://www.nist.gov/smartgrid/nistandthesmartgrid.cfm

Oracle. (2017). *JavaServer Pages Technology*. Retrieved from https://www.oracle.com/technetwork/java/javaee/jsp/index.html

PG&E. (2012, Mayo 4). *Pacific Gas And Electric.* Retrieved from http://www.pge.com/en/myhome/customerservice.page

Piattini. (1999). *Fundamentos de base de datos.* Mexico: Alfaomega.

Roadmap, I. S. (2012). *SMB Smart Grid Strategic Group*. Retrieved from SMB Smart Grid Strategic Group : http://www.iec.ch/smartgrid/donwloads/sg3_road_map.pdf

Sahin, D. (2011, mayo 12). *Smart Grid Tecnologies*.

Schauer, S. (2012). Implementing an Electric watt-hour metter. *Texas Instruments* , 1-32.

SEO. (2010, abril 2). *¿Que es el paradigma MVC?* Retrieved from http://www.dessarrolloMVC.com/que-es.el-paradigma-MVC

Sergeev, A., & Matulevicius, R. (2017). An Approach to Capture Role-Based Access Control Models from Spring Web Applications. *2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC).*

Shi, S., & Xu, C. (2016). Design and Implementation of a Role-Based Access Control for Categorized Resource in Smart Community Systems. *2016 7th International Conference on Cloud Computing and Big Data (CCBD).*

Sidney, U. d. (2012, Marzo 3). *Smart Grid* . Retrieved from http://www.ee.usyd.edu.au/about/smart-grid.html

Single-Phase Electricity Meter. (2013). *MQX*, 1-87.

Software, P. (2018, mayo). *Spring Security*. Retrieved from https://spring.io/projects/spring-security

Standards, N. I. (2010). *NIST Framework and roadmap for smart grid* .

Strzelecki, R. y. (2010). *Power Electronics in Smart Electrical Energy Networks.* Londres: Springer.

Suárez, E. (2009). *Diseño e Implementacion de base de datos relacionales.* México.

Systems, F. (2013, Marzo 3). *FREEDM Systems*. Retrieved from http://freedmnesu.edu/index/php

Theodore, P. (1974). *U.S Patent No. 3842208.*

Theodore, P. (1984). *Estados Unidos Patent No. 4241237.*

Tools, F. a. (2017). *Ajax in Bridge.Net*. Retrieved from https://forums.bridge.net/forum/community/help/4420-jquery-ajax-in-bridge-net

TWACS. (2012, Septiembre 12). *Aclara PLC*. Retrieved from http://www.aclaratech.com/AcaraPLC

University, N. S. (2011, junio 12). *Smart Grid Central*.

Vitoria, E. U. (2007, Mayo 6). *Patrón MVC*. Retrieved from http://www.lsi.vi.edu..es/mikell/docencia/Programacion

Węgrzynowicz, P. (2013). Performance antipatterns of one to many association in hibernate. *2013 Federated Conference on Computer Science and Information Systems*.

Wei, B. (2010). Applying Hibernate to persist Java3D virtual scene in Oracle. *Conference on Advanced Computer Control*.

Wollenberg, A. M. (2005). *Toward a Smart Grid : Power Delivery For The 21st Century* . US: IEEE .

Xia, C., Yu, G., & Tang, M. (2009). Efficient Implement of ORM (Object/Relational Mapping) Use in J2EE Framework: Hibernate. *Computational Intelligence and Software Engineering*.

Zao, L. (2012). *Implementation of 128 point FFT.* France: MRC60.

Zeev, C. (2012, Noviembre). *Narrowband PLC and the power line medium*. Retrieved from http://www.eetimes.com/document.asp

Zong Lin, R.-m. Z.-H. (2009). *Construction of Smart Gridand at information age.* Power System Technology.

Anexos.

Anexo A. Reconocimiento de la empresa.

EOS TECH

"Mejoramos tu mundo con innovación"

EOS TECH OTORGA EL SIGUIENTE

Reconocimiento

A: **ING. LUIS ÁNGEL ESPINA HERNÁNDEZ**

Por su participación en el desarrollo del Proyecto **Head End System**.

Se extiende una felicitación por la conclusión satisfactoria de su estancia profesional.

Ing. Juan González Salomón
DIRECTOR GENERAL
EOS TECH

Anexo B. Publicación realizada.

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

ACADEMIA JOURNALS

CONGRESO INTERNACIONAL DE INVESTIGACIÓN DE ACADEMIA JOURNALS.COM, CELAYA 2017

OTORGAN EL PRESENTE

CERTIFICADO

A

ING. LUIS ANGEL ESPINA HERNANDEZ
MSC AGUSTIN SANCHEZ ATONAL
MTRO. HIGINIO NAVA BAUTISTA

POR SU PARTICIPACIÓN CON LA PONENCIA TITULADA

DISEÑO E IMPLEMENTACIÓN DE UNA BASE DE DATOS RELACIONAL PARA UN HEAD END SYSTEM UTILIZANDO HIBERNATE ORM Y ORACLE

PUBLICADA EN EL PORTAL DE INTERNET
CELAYA.ACADEMIAJOURNALS.COM
VOLUMEN ONLINE CON ISSN 1946-5351 VOL. 9, NO. 6, 2017
E INDIZACIÓN EN FUENTE ACADÉMICA PLUS (EBSCO) Y
LIBRO DIGITAL EBOOK CON ISBN 978-1-939982-32-2, ONLINE.
LA CUAL FUE PRESENTADA EN EL
TECNOLÓGICO NACIONAL DE MÉXICO EN CELAYA
LOS DÍAS 8, 9 Y 10 DE NOVIEMBRE DE 2017, CELAYA, GUANAJUATO, MÉXICO.

DR. RAFAEL MORAS
EDITOR, ACADEMIAJOURNALS.COM
PROFESOR DE INGENIERÍA INDUSTRIAL Y ADMINISTRATIVA
ST. MARY´S UNIVERSITY, SAN ANTONIO, TX. EEUU

ING. ALEJANDRO ÁLVAREZ BÁRCENAS
COORDINADOR GENERAL DEL
CONGRESO INTERNACIONAL DE INVESTIGACIÓN
ACADEMIA JOURNALS, CELAYA 2017

N° 3121

ANAB

Cel1269

Anexo C. Cartas de liberación y satisfacción.

EOS TECH

Querétaro, Qro., a 02 de Agosto del 2017

Asunto: Constancia

MTRO. FELIPE PASCUAL ROSARIO AGUIRRE
DIRECTOR DEL INSTITUTO TECNOLÓGICO DE APIZACO
PRESENTE:

AT'N: DR. JOSÉ FEDERICO CASCO VÁZQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

Por medio del presente le envío un cordial saludo y aprovecho para hacerle constar que el Ing. Espina Hernández Luis Ángel, alumno de la Maestría en Sistemas Computacionales con número de control M10370700, de la Institución que usted destacadamente dirige, participó en el sub-proyecto:

"Diseño e implementación de una base de datos relacional utilizando *ORM programing technique, Spring Security e Hibernate para un head end system.*"

La etapa del proyecto realizada tuvo una duración de 6 meses y se desarrolló en las oficinas centrales de EOS TECH S.A. DE C.V. en la ciudad de Querétaro, Querétaro.

En virtud de que se han cubierto satisfactoriamente la fase del proyecto. Tenemos bien a dar constancia de que dicho trabajo realizado por el Ing. Espina Hernández Luis Ángel, cubre y satisface las expectativas planteadas al inicio de la fase de este proyecto.

Agradeciendo sus atenciones a la presente quedo de ustedes.

ATENTAMENTE

Ing. Juan González Salomón
Director General
EOS TECH S.A. DE C.V.

EOS TECH S.A de C. V.
Luis G. Malvaez No. 523 Col. Reforma Agraria 2da Sección.
Querétaro, Qro., México. C.P. 76086
Tel.: (442) 243 1331

www.eostech.mx
"Mejoramos tu mundo con innovación"

EOS TECH

Querétaro, Qro., a 02 de Agosto del 2017

Asunto: Constancia de satisfacción

MTRO. FELIPE PASCUAL ROSARIO AGUIRRE
DIRECTOR DEL INSTITUTO TECNOLÓGICO DE APIZACO
PRESENTE:

AT'N: DR. JOSÉ FEDERICO CASCO VÁZQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

Por medio del presente le envío un cordial saludo y aprovecho para hacerle constar que posterior a la recepción del sub-proyecto: **"Diseño e implementación de una base de datos relacional utilizando *ORM programing technique, Spring Security e Hibernate para un head end system*"** realizado por el Ing. Espina Hernández Luis Ángel con número de control M10370700, y dirigido por el M. Higinio Nava Bautista , alumno y profesor respectivos de la Maestría en Sistemas Computacionales, de la Institución que usted destacadamente dirige, **cubre y satisface las expectativas planteadas al inicio de la fase de este proyecto.**

Tenemos bien a dar constancia de que dicho trabajo realizado por el Ing. Espina Hernández Luis Ángel. Agradeciendo sus atenciones quedo de ustedes.

ATENTAMENTE

Ing. Juan González Salomón
Director General
EOS TECH S.A. DE C.V.

EOS TECH S.A de C. V.
Luis G. Malvaez No. 523 Col. Reforma Agraria 2da Sección.
Querétaro, Qro., México. C.P. 76086
Tel.: (442) 243 1331

www.eostech.mx
"Mejoramos tu mundo con innovación"

Querétaro, Querétaro a 19 de abril de 2017

ASUNTO: Carta de aceptación en Estancia Técnica

Mtro. Felipe Pascual Rosario Aguirre
Director - Instituto Tecnológico de Apizaco
Conurbado Apizaco - Tzompantepec s/n,
Col. Centro C.P.90300 Apizaco Tlaxcala
PRESENTE

Por este conducto me dirijo a usted para informarle la aceptación del C. Ing. Luis Ángel Espina Hernández alumno de la Maestría en Sistemas Computacionales con número de control M10370700 del Instituto Tecnológico de Apizaco, para realizar una Estancia Técnica en la Empresa EOS TECH S.A. de C.V. ubicada en Calle Luis G Malvaez número 523, Colonia Reforma Agraria en Querétaro, Querétaro. El objetivo general del proyecto en cual trabajará es el desarrollo de un Sistema del tipo *Head End* y un *Meter Data Management* System (MDMS) sin embargo los objetivos específicos del estudiante de Maestría se acota al siguiente entregable:

- Implementación de persistencia de datos para módulo de acceso a usuarios y registros de dispositivos inteligentes de un Head End System

Finalmente es importante indicar que el periodo que contempla la Estancia Técnica es del 07 de de febrero del 2017 al 04 de agosto del mismo año.
Sin otro particular, agradezco de antemano su atención. y quedo al pendiente para cualquier duda o aclaración.

Atentamente

JUAN GONZÁLEZ SALOMÓN
DIRECTOR GENERAL
EOS TECH S.A. de C.V.

EOS TECH S.A de C. V.
Luis G. Malvaez No. 523 Col. Reforma Agraria 2da Sección.
Querétaro, Qro., México. C.P. 76086
Tel.: (442) 243 1331

www.eostech.mx
"Mejoramos tu mundo con innovación"