



## **DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN**

### **CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA**

**TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN  
DEL TÍTULO DE INGENIERO EN SISTEMAS E  
INFORMÁTICA**

**TEMA: IMPLEMENTACIÓN DE UN SISTEMA WEB PARA  
GESTIÓN DE INVENTARIO DE ELEMENTOS  
HIDROMETEOROLÓGICOS PARA EL INSTITUTO  
NACIONAL DE METEOROLOGÍA E HIDROLOGÍA  
(INAMHI).**

**AUTOR: ESCOBAR TUCTA, JORGE**

**MENTOR**

**DIRECTOR: ING. SANCHO ARIAS,  
JOSÉ ALBERTO**

**SANGOLQUÍ**

**2016**

## CERTIFICADO



### DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN CARRERA DE INGENIERÍA EN SISTEMAS

#### CERTIFICACIÓN

Certifico que el trabajo de titulación, "**IMPLEMENTACIÓN DE UN SISTEMA WEB PARA GESTIÓN DE INVENTARIO DE ELEMENTOS HIDROMETEOROLÓGICOS PARA EL INSTITUTO NACIONAL DE METEOROLOGÍA E HIDROLOGÍA (INAMHI)**" realizado por el señor **ESCOBAR TUCTA JORGE MENTOR**, ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de las Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar al señor **ESCOBAR TUCTA JORGE MENTOR** para que lo sustente públicamente.

Ciudad, 18 de agosto del 2016

*José Alberto Sancho*

SANCHO ARIAS JOSE ALBERTO  
DIRECTOR

**AUTORÍA DE RESPONSABILIDAD**

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN  
CARRERA DE INGENIERÍA EN SISTEMAS

**AUTORÍA DE RESPONSABILIDAD**

Yo, **ESCOBAR TUCTA JORGE MENTOR**, con cédula de identidad N° 1719272336 declaro que este trabajo de titulación "**IMPLEMENTACIÓN DE UN SISTEMA WEB PARA GESTIÓN DE INVENTARIO DE ELEMENTOS HIDROMETEOROLÓGICOS PARA EL INSTITUTO NACIONAL DE METEOROLOGÍA E HIDROLOGÍA (INAMHI)**" realizado por el señor **ESCOBAR TUCTA JORGE MENTOR** ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaro que este trabajo es de mi autoría, en virtud de ello me declaro responsable del contenido, veracidad y alcance de la investigación mencionada.

Ciudad, 18 de agosto del 2016



\_\_\_\_\_  
ESCOBAR TUCTA JORGE MENTOR

C.C 1719272336

## AUTORIZACIÓN



### DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN CARRERA DE INGENIERÍA EN SISTEMAS

#### AUTORIZACIÓN

Yo, **ESCOBAR TUCTA JORGE MENTOR**, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar en la biblioteca Virtual de la institución el presente trabajo de titulación "**IMPLEMENTACIÓN DE UN SISTEMA WEB PARA GESTIÓN DE INVENTARIO DE ELEMENTOS HIDROMETEOROLÓGICOS PARA EL INSTITUTO NACIONAL DE METEOROLOGÍA E HIDROLOGÍA (INAMHI)**" cuyo contenido, ideas y criterios son de mi autoría y responsabilidad.

Ciudad, 18 de agosto del 2016



ESCOBAR TUCTA JORGE MENTOR

C.C 1719272336

## **DEDICATORIA**

*A mi sobrino Ezequiel quien es la nueva alegría de mi familia.*

*Jorge Escobar*

**AGRADECIMIENTO**

*A mi familia maravillosa Guadalupe, Mentor, Bárbara y Roberto, que no me han dejado caer a pesar de los problemas ofreciéndome todo el apoyo necesario para cumplir mis metas. A mis amigos Andrés Bustamante y Diego Hinojoza que me han brindado su conocimiento siendo un pilar fundamental para conseguir mi título de tercer nivel.*

Jorge Escobar

## ÍNDICE

<b>CERTIFICADO.....</b>	<b>II</b>
<b>AUTORÍA DE RESPONSABILIDAD.....</b>	<b>III</b>
<b>AUTORIZACIÓN .....</b>	<b>IV</b>
<b>DEDICATORIA .....</b>	<b>V</b>
<b>AGRADECIMIENTO .....</b>	<b>VI</b>
<b>ÍNDICE .....</b>	<b>VII</b>
<b>ÍNDICE DE TABLAS .....</b>	<b>IX</b>
<b>ÍNDICE DE FIGURAS .....</b>	<b>X</b>
<b>RESUMEN.....</b>	<b>XII</b>
<b>ABSTRACT .....</b>	<b>XIII</b>
<b>CAPÍTULO 1.....</b>	<b>1</b>
<b>INTRODUCCIÓN .....</b>	<b>1</b>
1.1. <i>Antecedentes .....</i>	1
1.2. <i>Planteamiento del Problema .....</i>	2
1.3. <i>Justificación .....</i>	10
1.4. <i>Objetivos.....</i>	12
a) <i>Objetivo General .....</i>	12
b) <i>Objetivos Específicos .....</i>	12
1.5. <i>Alcance .....</i>	12
<b>CAPÍTULO 2 .....</b>	<b>14</b>
<b>MARCO TEÓRICO.....</b>	<b>14</b>
2.1. <i>Metodología RUP .....</i>	14
2.1.1. <i>Principales Características: .....</i>	14
2.1.2. <i>Esfuerzo en actividades según fase del proyecto .....</i>	15
2.1.3. <i>Fases del ciclo de vida del rup: .....</i>	16
2.1.4. <i>La metodología RUP tiene 6 principios clave: .....</i>	17
2.1.5. <i>Disciplina de desarrollo de RUP .....</i>	18
2.1.6. <i>Disciplina de soporte RUP .....</i>	18
2.1.7. <i>Elementos de RUP .....</i>	19
2.2 <i>JEE.....</i>	19
2.2.1. <i>Introducción .....</i>	19
2.2.2. <i>Arquitectura multicapas de JAVA .....</i>	20
2.2.3. <i>Contenedores JAVA EE.....</i>	21
2.2.4. <i>Componentes JEE.....</i>	23
2.2.5. <i>Servicios estándar y tecnologías de soporte JEE .....</i>	24
2.3 <i>Framework Spring .....</i>	25
2.4 <i>Maven .....</i>	26
2.4.1. <i>Introducción .....</i>	26
2.4.2. <i>Objetivos de Maven .....</i>	27
2.4.3. <i>Pom .....</i>	29
2.4.4. <i>Coordenadas de maven.....</i>	31
2.4.5. <i>Relaciones del pom .....</i>	33
2.5 <i>Tipos de inventarios .....</i>	38
2.5.1. <i>Según su función:.....</i>	38
2.5.2. <i>Desde el punto de vista logístico: .....</i>	39
<b>CAPÍTULO 3 .....</b>	<b>40</b>
<b>FASE ANÁLISIS Y DISEÑO .....</b>	<b>40</b>
3.1 <i>Especificación de requerimientos de software .....</i>	40
3.1.1 <i>Introducción.....</i>	40

<b>3.1.2</b>	<i>Alcance y definiciones</i> .....	40
<b>3.1.3</b>	<i>Descripción general</i> .....	41
<b>3.1.4</b>	<i>Requerimientos específicos</i> .....	43
<b>3.1.5</b>	<i>Requerimientos funcionales</i> .....	44
<b>3.1.6</b>	<i>Requerimientos no funcionales</i> .....	49
<b>CAPÍTULO 4</b> .....		51
<b>FASE CODIFICACIÓN, IMPLEMENTACIÓN Y PRUEBAS</b> .....		51
<b>4.1</b>	<i>Desarrollo del prototipo</i> .....	51
<b>4.1.1</b>	<i>Tecnologías JAVA EE</i> .....	51
<b>4.1.2</b>	<i>Arquitectura del desarrollo</i> .....	52
<b>4.1.3</b>	<i>Paquetes de diseño</i> .....	57
<b>4.1.4</b>	<i>Vista de despliegue</i> .....	59
<b>4.1.5</b>	<i>Módulos del sistema</i> .....	60
<b>4.2</b>	<i>Pruebas</i> .....	90
<b>4.3</b>	<i>Plan de entrega de software</i> .....	93
<b>4.3.1</b>	<i>Introducción</i> .....	93
<b>4.3.2</b>	<i>Alcance</i> .....	94
<b>4.3.3</b>	<i>Procedimiento de entrega de software</i> .....	94
<b>CAPÍTULO 5</b> .....		95
<b>CONCLUSIONES Y RECOMENDACIONES</b> .....		95
<b>5.1</b>	<i>Conclusiones</i> .....	95
<b>5.2</b>	<i>Recomendaciones</i> .....	97
<b>REFERENCIAS BIBLIOGRAFIAS</b> .....		98

## ÍNDICE DE TABLAS

<b>Tabla 1. Definición del problema .....</b>	<b>9</b>
<b>Tabla 2. Posición del producto.....</b>	<b>11</b>
<b>Tabla 3. Módulos del Sistema .....</b>	<b>13</b>
<b>Tabla 4. Usuarios del sistema.....</b>	<b>42</b>
<b>Tabla 5. Descomposición de capas del sistema .....</b>	<b>53</b>
<b>Tabla 6. Caso de Prueba CP_F1.5 .....</b>	<b>90</b>
<b>Tabla 7. Caso de Prueba CP_F1.3 .....</b>	<b>90</b>
<b>Tabla 8. Caso de Prueba CP_F3.3 .....</b>	<b>91</b>
<b>Tabla 9. Caso de Prueba CP_F4.1 .....</b>	<b>92</b>
<b>Tabla 10. Caso de Prueba CP_RF4.2.3 .....</b>	<b>93</b>

## ÍNDICE DE FIGURAS

<b>Figura 1. Frecuencia de visitas a estaciones .....</b>	<b>3</b>
<b>Figura 2. Orden de Trabajo .....</b>	<b>4</b>
<b>Figura 3. Acta de Entrega Recepción.....</b>	<b>5</b>
<b>Figura 4. Acta de Entrega Recepción.....</b>	<b>6</b>
<b>Figura 5. Acta Recepción Entrega.....</b>	<b>7</b>
<b>Figura 6. Fases de RUP .....</b>	<b>15</b>
<b>Figura 7. Ciclo de vida de RUP .....</b>	<b>17</b>
<b>Figura 8. Historial de las versiones de Java .....</b>	<b>20</b>
<b>Figura 9. Arquitectura Multicapas .....</b>	<b>21</b>
<b>Figura 10. Contenedores JEE .....</b>	<b>22</b>
<b>Figura 11. Componentes JEE .....</b>	<b>23</b>
<b>Figura 12. Arquitectura de Spring .....</b>	<b>25</b>
<b>Figura 13. Elementos del POM .....</b>	<b>30</b>
<b>Figura 14. Ejemplo pom.xml.....</b>	<b>31</b>
<b>Figura 15. Empaquetado war .....</b>	<b>33</b>
<b>Figura 16. Dependencias del POM .....</b>	<b>34</b>
<b>Figura 17. Comando de instalación individual de dependencia .....</b>	<b>35</b>
<b>Figura 18. Diagrama General de casos de uso .....</b>	<b>44</b>
<b>Figura 19. Módulo Gestionar Usuarios.....</b>	<b>44</b>
<b>Figura 20. Módulo Gestionar las órdenes de trabajo de elementos .....</b>	<b>45</b>
<b>Figura 21. Gestionar elementos .....</b>	<b>45</b>
<b>Figura 22. Gestionar las solicitudes de elementos.....</b>	<b>46</b>
<b>Figura 23. Diagrama de clases .....</b>	<b>47</b>
<b>Figura 24. Diagrama físico .....</b>	<b>48</b>
<b>Figura 25. Diseño de la arquitectura del sistema .....</b>	<b>54</b>
<b>Figura 26. Fragmento de código de una clase bean .....</b>	<b>55</b>
<b>Figura 27. Configuración de faces servlet.....</b>	<b>56</b>
<b>Figura 28. Fragmento de código de un archivo xhtml.....</b>	<b>57</b>
<b>Figura 29. Paquetes de diseño .....</b>	<b>58</b>
<b>Figura 30. Diagrama físico de las redes que utiliza el sistema.....</b>	<b>59</b>
<b>Figura 31. Login de usuario .....</b>	<b>60</b>
<b>Figura 32. Lista de usuarios .....</b>	<b>60</b>
<b>Figura 33. Formulario para usuario nuevo .....</b>	<b>61</b>
<b>Figura 34. Formulario para edición de usuario .....</b>	<b>61</b>
<b>Figura 35. Lista de elementos .....</b>	<b>62</b>
<b>Figura 36. Formulario para elemento nuevo.....</b>	<b>62</b>
<b>Figura 37. Formulario para edición de elemento.....</b>	<b>63</b>
<b>Figura 38. Formulario para edición de propiedades de elemento.....</b>	<b>63</b>
<b>Figura 39. Lista de Bodegas .....</b>	<b>64</b>
<b>Figura 40. Formulario para bodega nueva.....</b>	<b>64</b>
<b>Figura 41. Formulario para edición de bodega.....</b>	<b>65</b>
<b>Figura 42. Eliminación de bodega .....</b>	<b>65</b>
<b>Figura 43. Lista de estados de elementos .....</b>	<b>66</b>
<b>Figura 44. Formulario para estado de elemento nuevo .....</b>	<b>66</b>
<b>Figura 45. Formulario para editar estado de elemento .....</b>	<b>67</b>
<b>Figura 46. Eliminación de estado de elemento .....</b>	<b>67</b>
<b>Figura 47. Catálogo de propiedades de elementos .....</b>	<b>68</b>
<b>Figura 48. Formulario para crear propiedad de elemento nueva .....</b>	<b>68</b>
<b>Figura 49. Formulario para editar propiedad de elemento .....</b>	<b>69</b>
<b>Figura 50. Eliminación de propiedad de elemento .....</b>	<b>69</b>
<b>Figura 51. Lista de Propiedades de elementos por propiedad .....</b>	<b>70</b>

<b>Figura 52. Formulario para crear un nuevo calor para una propiedad de elemento</b>	70
<b>Figura 53. Formulario para editar el valor de una propiedad de elemento</b>	71
<b>Figura 54. Eliminación del valor de una propiedad de elemento</b>	71
<b>Figura 55. Lista de clases de elementos</b>	72
<b>Figura 56. Formulario para crear una clase nueva</b>	72
<b>Figura 57. Formulario de edición de clase</b>	73
<b>Figura 58. Eliminación de clase</b>	73
<b>Figura 59. Lista de métodos de medición y observación</b>	74
<b>Figura 60. Formulario para crear método de medición u observación</b>	74
<b>Figura 61. Formulario de edición de método de medición u observación</b>	75
<b>Figura 62. Eliminación de método de medición u observación</b>	75
<b>Figura 63. Lista de tipos de elementos</b>	76
<b>Figura 64. Formulario para crear tipo de elemento nuevo</b>	76
<b>Figura 65. Formulario de edición de tipo de elemento</b>	77
<b>Figura 66. Eliminación de tipo de elemento</b>	77
<b>Figura 67. Lista de marcas de elementos</b>	78
<b>Figura 68. Formulario para crea una marca nueva</b>	78
<b>Figura 69. Formulario de edición de marca</b>	79
<b>Figura 70. Eliminación de marca</b>	79
<b>Figura 71. Lista de líneas</b>	80
<b>Figura 72. Formulario de creación de línea nueva</b>	80
<b>Figura 73. Formulario de edición de línea</b>	81
<b>Figura 74. Eliminación de línea</b>	81
<b>Figura 75. Formulario para crear una nueva solicitud de elementos</b>	82
<b>Figura 76. Lista de estación desplegable</b>	82
<b>Figura 77. Lista de elementos disponibles para agregar a la solicitud</b>	83
<b>Figura 78. Solicitud de elementos llena</b>	83
<b>Figura 79. Lista de solicitudes</b>	84
<b>Figura 80. Solicitud seleccionada</b>	84
<b>Figura 81. Elementos pedidos en una solicitud</b>	85
<b>Figura 82. Lista de estados de solicitudes</b>	85
<b>Figura 83. Formulario para crear estado de solicitud nuevo</b>	86
<b>Figura 84. Formulario de edición de estado de solicitud</b>	86
<b>Figura 85. Formulario para crear una orden de trabajo nueva</b>	87
<b>Figura 86. Formulario para escoger elementos y tareas que se requieren realizar</b>	87
<b>Figura 87. Orden de trabajo llena</b>	88
<b>Figura 88. Lista de órdenes de trabajo</b>	88
<b>Figura 89. Elementos vinculados a orden de trabajo seleccionada</b>	89

## RESUMEN

El Instituto Nacional de Meteorología e Hidrología para el cual se realizó el proyecto, se enfrentaba a la carencia de información acerca de la existencia, localización y disponibilidad de los elementos o instrumentos necesarios para dar mantenimiento a las 121 estaciones en todo el Ecuador. Los ingenieros del INAMHI tenían que buscar físicamente los elementos en todas las instalaciones del instituto lo cual producía retrasos en los mantenimientos. El objetivo del presente proyecto es desarrollar un sistema web que permita optimizar el proceso de gestión de elementos para el INAMHI el cual pueda dar a conocer la localización de los mismos. La metodología de software RUP fue utilizada en el análisis y diseño, la herramienta Maven se encargó de gestionar las librerías y dependencias, con los servicios que ofrece Hibernate se pudieron manejar las relaciones de los objetos con la base de datos y en cuanto al desarrollo de la interfaz de usuario se manejó la tecnología JSF. El sistema web permite conocer el departamento, bodega o estación del INAMHI en el cual se encuentran los artículos. La aplicación además sirve para gestionar solicitudes de instrumentos y órdenes de trabajo las cuales ayudan a llevar un registro de los artículos que entran a reparaciones en los departamentos de mecánica o electrónica o calibraciones en el departamento de metrología. Con este proyecto se estandarizó la gestión de los elementos y órdenes de trabajo, como consecuencia se redujeron considerablemente los retrasos y las suspensiones de visitas técnicas a las estaciones meteorológicas.

### PALABRAS CLAVE:

- **SISTEMA**
- **WEB**
- **SPRING**
- **MAVEN**
- **RUP**
- **INVENTARIOS.**

## **ABSTRACT**

The National Institute of Meteorology and Hydrology for which the project has been developed, is facing the lack of information about the existence, location and availability of the necessary elements for give maintenance to 121 stations throughout Ecuador, so INAMHI engineers have to physically look for items in all the institute which causes delays in servicing. The aim of this project is to develop a web system that optimizes the management process elements for INAMHI which allows to know the location of these elements. Software methodology RUP was used for the analysis and design, the Maven tool was in charge to manage libraries and dependencies, with the services provided by Hibernate the system can manage relationships of objects with the database, to manage the infrastructure, the Spring framework was employed, the JSF technology was handled to develop the user interface. The web system allows to know the department, warehouse or INAMHI station where the elements are. The application also allows to manage requests for items and work orders which help keep track of the elements when they enter for repairs in the mechanical or electronic departments or for calibrations in the metrology department. With this project the elements and work orders management were standardized, As a consequence of this, delays and suspensions of technical visits to weather stations were reduced considerably.

### **KEY WORDS:**

- **SISTEM**
- **WEB**
- **SPRING**
- **MAVEN**
- **RUP**
- **INVENTARIOS.**

## CAPÍTULO 1

### INTRODUCCIÓN

#### 1.1. Antecedentes

El inventario es una pieza primordial en cualquier empresa. Tener un registro de los productos existentes, disponibles y en qué área de la organización se encuentran, es básico para poder tener todo organizado y que no haya pérdidas de recursos materiales o tiempo de las personas. La adecuada estructura y gestión de inventarios es uno de los factores primordiales que incurren en el desempeño empresarial y en los dividendos que se obtienen. Por lo tanto es esencial para las organizaciones basarse en un inventario bien gestionado y controlado. La orientación en este campo tiene que ser el de mantenerse un nivel óptimo para no producir gastos no planificados. El seguimiento y control de inventarios es una actividad modular en una empresa ya que al tener una visibilidad confiable del mismo se puede tomar mejores decisiones además de reducir costos al no cargar con excesos o faltantes en artículos y por lo tanto ofrecer un mejor servicio al cliente. Además, el tener un inventario que esté en un nivel óptimo puede apoyar a liberar flujo de efectivo.

En la actualidad el gestionar los inventarios mediante aplicaciones computacionales en grandes y pequeñas empresas se ha convertido en una buena práctica de la cual no se puede prescindir ya que brinda rapidez, exactitud y en muchos casos establece la forma en que se manejan los procesos que tienen que ver con los inventarios.

La administración y control de inventarios requiere un tratamiento exhaustivo ya que tiene un gran impacto en todas las áreas operativas de las empresas y a su vez es un aspecto primordial de la administración ya que cuando no está controlado implica un alto costo y requiere una mayor inversión económica.

Normalmente el mercado da una pelea muy dura y esto obliga a las organizaciones a no permitirse tener su dinero estancado en forma de mercadería en su inventario, tampoco pueden darse el lujo de no estar capacitados para dar un óptimo servicio al consumidor al quedarse sin productos. La meta es conseguir esa armonía entre la oferta y la demanda, también es importante tener confiabilidad en los momentos de ingreso de artículos de su abastecedor como cuando se entrega a sus compradores.

Durante los últimos años la web pasó de ofrecer información estática a brindar un conjunto variado de servicios. Un avance significativo en la tecnología web fue la plataforma Java, la novedad de este lenguaje fue que es independiente de la plataforma cliente.

Hoy en día, la tecnología Java ha cobrado mucha importancia en el ámbito de internet gracias a su plataforma JEE (Java Enterprise Edition) la cual soporta aplicaciones distribuidas, toma ventajas de las tecnologías existentes y en desarrollo simplifica el proceso a través de un modelo de aplicaciones basados en componentes. Tal es el caso de Spring el cual está fundamentado en la reutilización del software, por tal motivo el código se hace más fácil de probar, se organizan los objetos de la capa intermedia fácilmente. Spring está diseñado para que existan las menores dependencias posibles (casi todos los objetos de negocio NO dependen del framework). Spring no solo ha popularizado algunas mejores prácticas del uso del núcleo de java, ahora está influenciando el desarrollo del mismo java.

El presente proyecto será implementado en el INAMHI, creado como una necesidad y un derecho fundamental de la comunidad ecuatoriana, con la capacidad y la obligación de suministrar información vital sobre el tiempo, el clima y los recursos hídricos del pasado, presente y futuro, que necesita conocer el país para la protección de la vida humana y los bienes materiales. Es una Institución con representación nacional e internacional, miembro de la Organización Meteorológica Mundial (OMM), organización intergubernamental especializada de las Naciones Unidas para la Meteorología (el tiempo y el clima), la Hidrología Operativa y las ciencias conexas.

## **1.2. Planteamiento del Problema**

El INAMHI necesita realizar la gestión de redes de estaciones automáticas de aplicación hidrológica y meteorológica (instalaciones destinadas a medir y registrar regularmente diversas variables meteorológicas). Esto incluye procedimientos los cuales deben hacerse de acuerdo a los manuales técnicos de la OMM en los cuales una de las principales indicaciones es que las estaciones meteorológicas automáticas deben inspeccionarse por lo menos una vez cada seis meses.

c) los métodos de observación y los procedimientos de reducción de las observaciones se aplican de manera uniforme; y
d) los observadores reúnen las cualidades exigidas de competencia para realizar sus tareas.
3.1.7 Todas las estaciones sinópticas terrestres deberían inspeccionarse por lo menos una vez cada dos años.
3.1.8 Las estaciones meteorológicas agrícolas y las estaciones especiales deberían inspeccionarse por lo menos una vez al año.
3.1.9 Las estaciones climatológicas principales deberían inspeccionarse por lo menos una vez al año; las estaciones climatológicas ordinarias y las de medición de la precipitación por lo menos una vez cada tres años. De ser posible, durante la estación de invierno, deberían realizarse ocasionalmente las inspecciones pertinentes.
3.1.10 Las estaciones meteorológicas automáticas deberían inspeccionarse por lo menos una vez cada seis meses.
3.1.11 En las estaciones marítimas, los barómetros deberían controlarse por lo menos dos veces al año con referencia a un barómetro patrón.

### Figura 1. Frecuencia de visitas a estaciones

Fuente: (Manual del Sistema Mundial de Observación, 2014)

Por lo cual, equipos técnicos llamados comisiones deben ir a las 121 estaciones semestralmente a revisar, calibrar o cambiar elementos si es necesario. Para lo cual, existe un plan anual de visitas a las estaciones. El objetivo de dicha preparación de comisiones no se cumple principalmente porque al momento que la comisión va a salir a la visita no se dispone de los elementos necesarios.

El INAMHI no cuenta con un sistema web que permita a los ingenieros designados a las comisiones conocer la existencia, localización ni disponibilidad de elementos necesarios para llevar a la revisión de una estación, por tal motivo tienen que solicitar elementos en el Departamento de Operación y Mantenimiento de la Red Automática (OMRA) cuyos empleados tienen que buscar físicamente los elementos en los siguientes lugares:

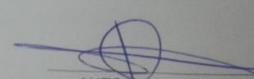
- Bodega de OMRA
- Laboratorio de Metrología
- Laboratorio de Electrónica

- Laboratorio de Mecánica
- Cualquiera de las 121 estaciones

De existir el elemento buscado puede tener varios estados, puede ser reparado, nuevo, calibrado, etc. Cuando se realiza la solicitud de dicho elementos no se sabe su estado actual, por lo que se puede recibir un elemento que haya sido dado de baja o simplemente tenga algún inconveniente técnico.

Los elementos no poseen numeración ni un código que los identifiquen por lo cual no se puede llevar una contabilidad de los mismos e inclusive teniendo el elemento en la mano no se puede asegurar su identidad, mucho menos saber su historial.

Los departamentos de metrología, mecánica y electrónica del INAMHI reciben de OMRA elementos que necesitan ser reparados o calibrados. Cuando un elemento ingresa a uno de estos departamentos se llena una orden de trabajo. Ejemplo:

INSTITUTO NACIONAL DE METEOROLOGÍA E HIDROLOGÍA "INAMHI" ORDEN DE TRABAJO IDT 2014	
FECHA: 22/01/2014	CODIGO: LM-OT- 00 7
SOLICITANTE: Ing. Javier Acosta	AREA: IDT
TRABAJO A REALIZAR: Construcción Estación hidrológica para estación tan sensor tipo radar (puente PAYANTINO)	
MATERIAL EMPLEADO: plancha acero 1/8", 3/8", pernos soldadora mig	MATERIAL EXTERNO:
FECHA DE ENTREGA: 05/02/2014	
OBSERVACIONES:	
 FIRMA DEL TECNICO JAVIER BRUGUERA	
 AUTORIZADO POR	
 FIRMA SOLICITANTE	

**Figura 2. Orden de Trabajo**

Fuente: (INAMHI, 2014)

 <i>"Saber para Ser"</i>		
<b>ACTA DE ENTREGA RECEPCIÓN</b>		
<p>El Centro de Investigación Energías Alternativas y Ambiente de la ESPOCH el lunes 05 de octubre en la ciudad de Quito del 2015, entrega al Inamhi los Sensores pertenecientes a una estación Meteorológica del Centro de Investigación Energías Alternativas y Ambiente ESPOCH para su respectiva calibración, con las siguientes especificaciones.</p>		
NOMBRE	MODELO	SERIE
Caja Mena	Sutron / Nema 4	X 1978261 ✓
Conversor	2/24Vdc	735534 ✓✓
Wireless		✓
Sensor de Presión Barométrica	Sutron / 5600-0120-AP	✓ ✓
Data Logger AN2201C 3484042	Sutron / 8310-0	J044 ✓ X
Sensor de Temperatura y Humedad Relativa	Sutron / 5600-0310-AP	✓ X
Sensor de Velocidad y Dirección de Viento	Sutron / Yung / 05103	130763 X
2 Piranómetros	SR110	7958 / 7898
Pluviómetro	5600-0525-2	1303408 ✓ 9
2 Sensores de Perfil de Temperatura de Suelo	QMT107	J42127 ✓ J42123 ✓

**ENTREGADO POR:**

Dr. Rigoberto Muñoz  
DIRECTOR CENTRO DE ENERGÍAS  
ALTERNATIVAS Y AMBIENTE-ESPOCH

**RECIBIDO POR:**

Ing. Kleber Rodriguez  
INAMHI

**Temperatura -40 -60° C, 0,1  
Humedad 0.100 %HC**

**± 2% 0.1 mm**

**Entrega Viernes 30 de Octubre 2015 .**

**Figura 3. Acta de Entrega Recepción**

Fuente: (INAMHI, 2015)

	<i>"Saber para Ser"</i>	
<b>ACTA DE ENTREGA RECEPCIÓN</b>		
<p>El Centro de Investigación Energías Alternativas y Ambiente de la ESPOCH el jueves 26 de Noviembre en la ciudad de Quito del 2015, entrega al Inamhi los Sensores pertenecientes a una estación Meteorológica del Centro de Investigación Energías Alternativas y Ambiente ESPOCH para su respectiva calibración, con las siguientes especificaciones.</p>		
NOMBRE	MODELO	SERIE
2 Piranometros	SR11	7949 7975 / <i>sin conectar</i>
1 Sensor de Perfil de Temperatura de Suelo	QMT107	J37301 /

**ENTREGADO POR:**

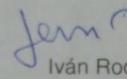
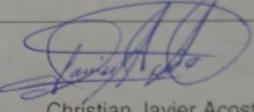
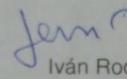
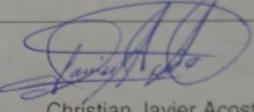
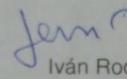
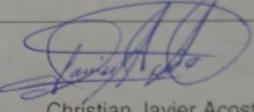
Ing. Gladys Janneth Urquiza Buenano  
 ESPECIALISTA EN INVESTIGACION,  
 PROYECTOS Y TRANSFERENCIA DE  
 TECNOLOGIAS 2 CEAA-ESPOCH

**RECIBIDO POR:**

Ing. Kleber Rodriguez  
 INAMHI

**Figura 4. Acta de Entrega Recepción**

Fuente: (INAMHI, 2015)

 INSTITUTO NACIONAL DE METROLOGÍA, GESTIÓN Y MEDIOAMBIENTE		<a href="http://www.inamhi.gob.ec">www.inamhi.gob.ec</a> Quito N36-14 y Corea 593 2 3971100 Oficina Encuentro																																	
<b>ACTA RECEPCIÓN-ENTREGA N°</b> <span style="border: 1px solid black; padding: 2px;">  </span> <span style="border: 1px solid black; padding: 2px;">  </span> <span style="border: 1px solid black; padding: 2px;">  </span>																																			
En la ciudad de Quito, a <u>21</u> del mes de <u>Abril</u> del 2015, EL LABORATORIO DE METROLOGÍA DE INAMHI, recibe del Ing. <u>Javier Acosta</u> los equipos detallados en la tabla 1, para su calibración.																																			
Tabla 1: Descripción de equipos entregados al Laboratorio de Metrología del INAMHI.																																			
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="3">CAMPOS PARA LLENAR EL SOLICITANTE</th> <th colspan="3">LABORATORIO</th> </tr> <tr> <th>Cantidad</th> <th>Descripción</th> <th>Marca</th> <th>Serie</th> <th>Recibe</th> <th>Entrega</th> </tr> </thead> <tbody> <tr> <td>4</td> <td>Sensor de temperatura</td> <td>Logotronic</td> <td>-----</td> <td></td> <td>8/05/2015</td> </tr> <tr> <td>6</td> <td>Sensor de Humedad relativa</td> <td>Logotronic</td> <td>-----</td> <td></td> <td>8/05/2015</td> </tr> <tr> <td>6</td> <td>Pluviómetros de balancín</td> <td>Young</td> <td>-----</td> <td></td> <td>8/05/2015</td> </tr> </tbody> </table>						CAMPOS PARA LLENAR EL SOLICITANTE			LABORATORIO			Cantidad	Descripción	Marca	Serie	Recibe	Entrega	4	Sensor de temperatura	Logotronic	-----		8/05/2015	6	Sensor de Humedad relativa	Logotronic	-----		8/05/2015	6	Pluviómetros de balancín	Young	-----		8/05/2015
CAMPOS PARA LLENAR EL SOLICITANTE			LABORATORIO																																
Cantidad	Descripción	Marca	Serie	Recibe	Entrega																														
4	Sensor de temperatura	Logotronic	-----		8/05/2015																														
6	Sensor de Humedad relativa	Logotronic	-----		8/05/2015																														
6	Pluviómetros de balancín	Young	-----		8/05/2015																														
Los sensores serán entregados calibrados <u>el 8 de mayo 2015</u> con sus respectivos certificados.																																			
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="2" style="text-align: center;">Recepción</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">           Iván Rodríguez          Laboratorio de Metrología       </td> <td style="text-align: center;">           Christian Javier Acosta          Institución       </td> </tr> <tr> <td colspan="2">         Observación: De no existir pluviómetros marca Young completar con pluviómetros marca Texas       </td> </tr> <tr> <th colspan="2" style="text-align: center;">Entrega</th> </tr> <tr> <td style="text-align: center;">         Iván Rodríguez          Laboratorio de Metrología       </td> <td style="text-align: center;">         Christian Javier Acosta          Institución       </td> </tr> <tr> <td colspan="2">         Observación:       </td> </tr> </tbody> </table>						Recepción		 Iván Rodríguez Laboratorio de Metrología	 Christian Javier Acosta Institución	Observación: De no existir pluviómetros marca Young completar con pluviómetros marca Texas		Entrega		Iván Rodríguez Laboratorio de Metrología	Christian Javier Acosta Institución	Observación:																			
Recepción																																			
 Iván Rodríguez Laboratorio de Metrología	 Christian Javier Acosta Institución																																		
Observación: De no existir pluviómetros marca Young completar con pluviómetros marca Texas																																			
Entrega																																			
Iván Rodríguez Laboratorio de Metrología	Christian Javier Acosta Institución																																		
Observación:																																			

**Figura 5. Acta Recepción Entrega**

Fuente: (INAMHI, 2015)

Estos documentos no se los ingresa a una base de datos digital, solo se guarda el papel en carpetas físicas. Esta falta de conocimiento del estado del elemento impide tener una mejor planeación para tomar decisiones pertinentes para mejorar la eficiencia de los servicios que ofrecen los departamentos de electrónica, mecánica y metrología.

Todo este proceso genera retrasos en la salida de comisiones ya planificadas para el mantenimiento de las estaciones lo que produce pérdida de tiempo, recursos materiales y financieros.

Por estos retrasos que duran varios días no se instala a tiempo el elemento en las estaciones lo que produce pérdida de información lo cual demora las predicciones de

información fundamental; principalmente en alertas tempranas que pueden salvar muchas vidas, reducen los daños materiales y protegen el medio ambiente.

**Tabla 1.****Definición del problema**

El problema de	No cumplir con el plan de mantenimiento de estaciones automáticas debido a que el INAMHI no cuenta con un sistema web que permita a los ingenieros designados a las comisiones conocer la existencia, localización ni disponibilidad de los elementos necesarios para llevar a la revisión de la estación, por tal motivo tienen que solicitar elementos en el Departamento de Operación y Mantenimiento de la Red Automática (OMRA) cuyos empleados tienen que buscar físicamente los elementos en varios departamentos del instituto.
Que afecta a	Las comisiones, las cuales sufren retrasos en salidas ya planificadas para el mantenimiento de las estaciones lo que produce pérdida de tiempo, recursos materiales y financieros.
El impacto de ello es	Por estos retrasos que duran varios días no se instala a tiempo los elementos en las estaciones lo que produce pérdida de información lo cual demora las predicciones de información fundamental; principalmente en alertas tempranas que pueden salvar muchas vidas, reducen los daños materiales y protegen el medio ambiente.
Una solución exitosa debería	<p>Implementar un sistema web para gestión de inventario de elementos para poder conocer la existencia, localización y disponibilidad de los elementos.</p> <p>Este sistema debería permitir a los ingenieros designados para las comisiones hacer solicitudes de elementos con lo cual se realizará una reserva de los elementos, los mismos que no podrían ser utilizados para ningún otro propósito.</p> <p>El Sistema Web permitiría a la persona que gestiona el inventario agregar elementos nuevos y quitar los que hayan sido dados de baja, además de establecer la localización de cada uno, lo que ayudará a los ingenieros electrónicos a conocer la calidad de cada elemento. El sistema evitaría que los solicitantes reciban elementos dados de baja o que se encuentren en alguna estación y así se desperdicie tiempo y recursos en resolver estos problemas.</p>

### 1.3. Justificación

Mediante el desarrollo de un sistema web para gestión de inventario de elementos hidrometeorológicos para el INAMHI se podrá conocer la existencia, localización y disponibilidad de estos. Los elementos se marcarán mediante código QR, este código se ingresará al sistema mediante un lector de código QR tipo pistola. Con el cual se facilitara el registro de elementos y ya no habrá el problema de que algunos tienen código y otros no tienen.

Los ingenieros designados para las comisiones podrán hacer solicitudes de elementos con lo cual se realizará una reserva, los mismos que no podrán ser utilizados para ningún otro propósito.

El Sistema Web permitirá a la persona que gestiona el inventario en OMRA agregar elementos nuevos y quitar los que hayan sido dados de baja, además de establecer la localización de cada uno. El sistema evitará que los solicitantes reciban elementos dados de baja o que se encuentren en alguna estación y así se desperdicie tiempo y recursos en resolver estos problemas, esto impedirá que las comisiones programadas anualmente sufran retrasos por no encontrar elementos.

Con la información generada por el sistema web, se podrá planificar el tratamiento a los elementos según su estado, enviarlos a arreglar o solicitar presupuesto para comprar nuevos.

El sistema web ayudará al departamento de mecánica, electrónica, metrología y OMRA a seguir procesos y buenas prácticas definidas por la OMM en cuanto al mantenimiento de las estaciones y elementos, esto favorecerá a la conservación de las estaciones meteorológicas e hidrológicas en óptimo estado para que los datos receptados sean confiables y de buena calidad, para el uso de los diferentes departamentos del INAMHI que procesan dicha información como son:

- Investigación
- Meteorología
- Pronóstico del tiempo
- Hidrología

Los departamentos de Electrónica, Metrología y Mecánica ya no usaran papel ni carpetas físicas para almacenar las hojas de trabajo. Ahora llenarán órdenes de trabajo en el sistema web mediante un navegador y sus respectivos datos se almacenaran en un servidor del INAMHI.

**Tabla 2.**

**Posición del producto**

Para	Instituto Nacional de Meteorología e Hidrología INAMHI
Quien	Requiere cumplir el plan anual de mantenimiento de estaciones automáticas del Ecuador.
El Sistema web para gestión de inventario de elementos hidrometeorológico para el Instituto Nacional de Meteorología e Hidrología.	Es un sistema web desarrollado en java con JEE y Spring.
Que	Es un software que se realizará con el objetivo de optimizar la gestión de elementos.
Diferente a	Seguir sin gestionar los elementos, lo cual continuaría afectando al funcionamiento de las estaciones.
Mi producto	Es un sistema moderno construido específicamente para resolver los inconvenientes que actualmente se tienen en el mantenimiento de estaciones y manejo de elementos.

#### **1.4. Objetivos**

##### **a) Objetivo General**

Desarrollar un sistema web que permita optimizar el proceso de gestión de elementos para el Instituto Nacional de Meteorología e Hidrología INAMHI.

##### **b) Objetivos Específicos**

- i. Optimizar y automatizar los procesos de gestión de elementos que posee el INAMHI.
- ii. Utilizar las tecnologías JEE con Spring y la metodología RUP para la construcción del sistema.
- iii. Implementar e implantar el sistema en las instalaciones del INAMHI.

#### **1.5. Alcance**

El proyecto comenzará con una investigación exhaustiva acerca del framework de desarrollo de software Spring con Ajax. El sistema web estará compuesto de los siguientes módulos:

**Tabla 3.****Módulos del Sistema**

<b>MODULO</b>	<b>DESCRIPCIÓN</b>	<b>USUARIO</b>
<b>Gestión de Usuarios</b>	Ingreso al sistema, creación de nuevos usuarios, edición y consulta de usuarios existentes.	Administrador de bodega.
<b>Gestión de elementos</b>	Ingreso, edición, consulta y actualización de inventario.	Administrador de bodega.
<b>Gestión de solicitudes de elementos</b>	El ingeniero que sea designado en una comisión para la instalación o mantenimiento de una estación será capaz de solicitar elementos a través de una interfaz web con la cual se realizará una reserva de los elementos pedidos, a su vez, el personal de bodega tendrá acceso a esas solicitudes y podrá aprobarlas en su totalidad, negarlas o aprobarlas parcialmente para el despacho de los mismos.	Administrador de bodega, Personal de OMRA.
<b>Gestión de órdenes de trabajo de elementos</b>	Permite conocer el estado del trabajo sobre el elemento. Genera órdenes de trabajo para el cambio del estado de los elementos y su respectivo ingreso al laboratorio de metrología, electrónica y mecánica.	Administrador de la bodega, personal de metrología, personal de mecánica, personal de electrónica.

El sistema web no incluirá ningún otro procedimiento de la empresa que no tenga que ver con inventarios de elementos.

## CAPÍTULO 2 MARCO TEÓRICO

### 2.1. Metodología RUP

El presente proyecto, que lleva por título “Implementación de un sistema web basado en JEE con Spring para gestión de inventario de elementos hidrometeorológicos para el INAMHI”, se realizará basándose en la metodología RUP (Rational Unified Process), tomando de ella aquellas características que permitan desarrollar el proyecto de la mejor manera y que se ajusten a las condiciones y tamaño del mismo.

#### Rational Unified Process (RUP)

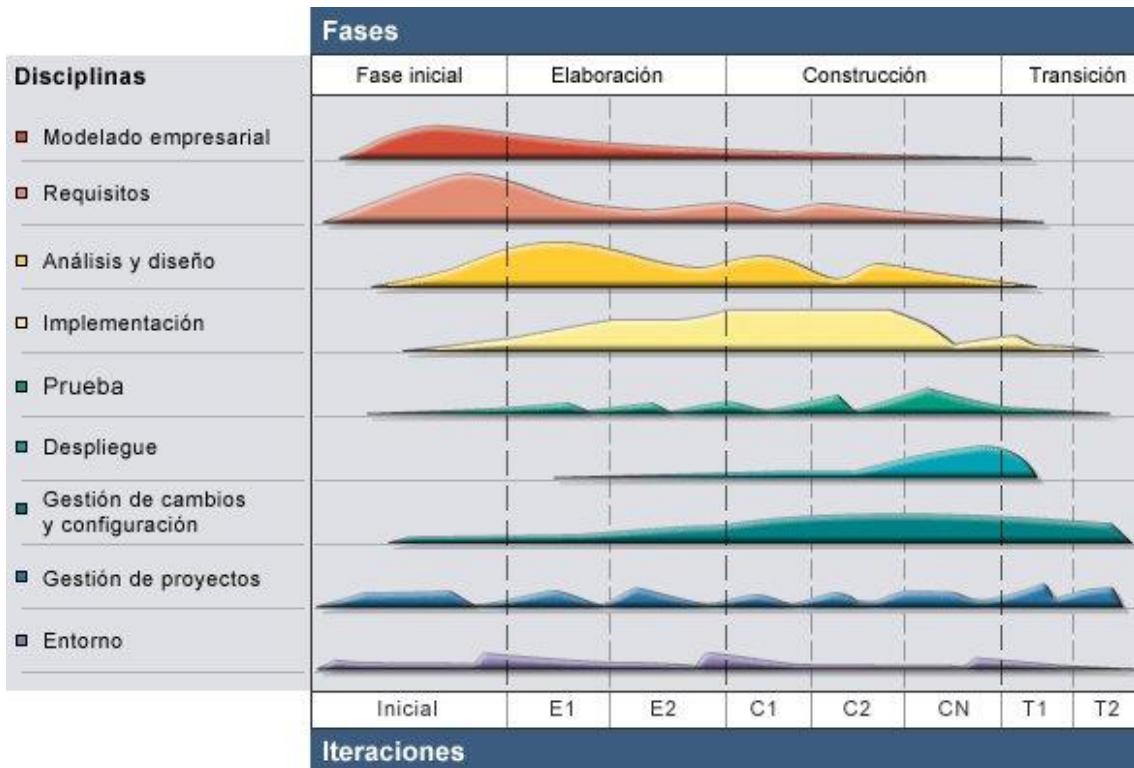
En esta metodología la meta es proporcionar un producto software. Consiste en llevar una documentación exhaustiva de todo el proyecto en la cual se basará la construcción del mismo y cumplir un plan de proyecto establecido en la fase inicial del desarrollo del proyecto. Es un proceso de desarrollo de software el cual utiliza el lenguaje unificado de modelado UML, es una metodología estándar y actualmente la más empleada para el análisis, implementación y documentación de aplicaciones orientadas a objetos. RUP constituye un grupo de metodologías amoldable al contexto y exigencias de cada empresa.

Describe la manera de aplicar enfoques para el desarrollar un producto software, tomando en cuenta varios pasos para su ejecución. Prioriza la producción y mantenimiento de prototipos de su sistema.

#### 2.1.1. Principales Características:

- Tiene una forma ordenada de establecer tareas y responsabilidades (quién hace qué, cuándo y cómo)
- Pretende hacer uso de las mejores prácticas que tiene la Ingeniería de Software
- Iterativo e incremental
- Administra los requisitos
- Usa una arquitectura basada en componentes
- Tiene control de cambios
- Pretende hacer un modelado visual del software
- Realiza verificación de la calidad del software

RUP es un producto de IBM. Trabaja centrado en la arquitectura la cual se define en sus artefactos y está guiado por los casos de uso y demás diagramas UML. Estos artefactos son los productos tangibles de la construcción. Algunos diagramas pueden ser: el modelo de casos de uso, el código fuente, etcétera. En los documentos de RUP una persona puede desempeñar uno o más roles en las diferentes partes del proceso de construcción del software.



**Figura 6. Fases de RUP**

Fuente: (Fowler, 2015)

### 2.1.2. Esfuerzo en actividades según fase del proyecto

La implementación del ciclo de vida de la metodología RUP se realiza en espiral. Esto se realiza acomodando los elementos de cada ciclo ordenadamente en cada secuencia. Esto queda organizado en forma de tareas dentro de fases e iteraciones las cuales forman parte del ciclo de vida.

El proceso de la metodología RUP está dividido en cuatro fases, dentro de estas fases se tienen que realizar entre dos o más iteraciones dependiendo el cumplimiento de los requerimientos del cliente, según esto se da mayor o menor prioridad en las actividades de la siguiente iteración.

### **2.1.3. Fases del ciclo de vida del rup:**

#### **a) Fase de Inicio**

Para esta fase se plantean cumplir con varias tareas como determinar y concretar con los patrocinadores el alcance del sistema, determinar los peligros asociados al proyecto, exponer una percepción genérica de la arquitectura del proyecto y establecer la planificación de las fases y las iteraciones posteriores.

#### **b) Fase de elaboración**

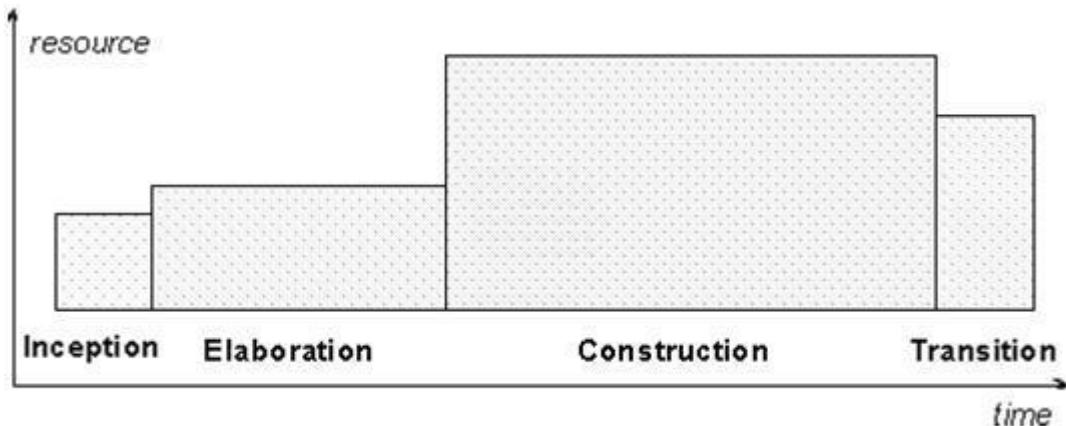
La elaboración de los casos de uso es primordial para el resto de construcción del sistema, estos se los realiza en esta fase, los mismos que serán una base para ir construyendo los módulos del proyecto. Se tiene que detallar cada caso de uso y luego un análisis de todo lo que abarca el problema, dando como resultado una solución preliminar.

#### **c) Fase de Construcción**

En esta fase se debe hacer más específicos los requerimientos pendientes con el objetivo de perfeccionar las funcionalidades del sistema. También se deben gestionar los cambios tomando en cuenta evaluaciones realizadas por usuarios, con esto hacer mejoras en el sistema.

#### **d) Fase de Transición**

Con el objetivo de asegurar que el producto software esté disponible para los usuarios finales, en esta fase se corrigen los errores y defectos descubiertos en las pruebas de aceptación, se prepara a los futuros usuarios y se les entrega el soporte técnico necesario. Se realiza una revisión en la cual se cumpla una relación de concordancia del producto con las especiaciones proporcionadas por los encargados del proyecto.



**Figura 7. Ciclo de vida de RUP**

Fuente: (Fowler, 2015)

#### 2.1.4. La metodología RUP tiene 6 principios clave:

##### a) Adaptación del proceso

El proceso tiene que llegar a adaptarse a las propiedades de la empresa para la que se está construyendo el software.

##### b) Balancear prioridades

Las prioridades del sistema deben establecerse de tal manera que complazcan a todas las personas que están invirtiendo en el proyecto.

##### c) Colaboración entre equipos

Se debe manejar una comunicación frecuente y las partes deben ser fáciles de contactar. Esto con el objetivo de acordar requerimientos, desarrollar evaluaciones, realizar planes, sacar resultados, etc.

##### d) Demostrar valor iterativamente

Las iteraciones se deben entregar, por lo menos internamente en sus respectivas etapas. Todas las iteraciones deben ser evaluadas en cuanto a calidad y estabilidad del sistema y se debe además examinar los criterios y proposiciones de los inversores.

##### e) Elevar el nivel de abstracción

En medida de lo posible reutilizar métodos y querys.

**f) Enfocarse en la calidad**

Se debe comprobar que la calidad del software sea la mejor, en cada etapa de la producción.

**2.1.5. Disciplina de desarrollo de RUP**

Establece las fases que se van a llevar a cabo a lo largo del proyecto de construcción del software.

**a) Ingeniería o modelado del negocio**

Examinar y comprender lo que la empresa para el cual se está construyendo el software necesita.

**b) Requisitos**

Proporcionar una referencia desde donde se pueda valorar los costos y cuanto va a durar la construcción del proyecto.

**c) Análisis y diseño**

Realizar un sistema automatizado basándose en los requerimientos examinados anteriormente y construir una arquitectura para el proyecto.

**d) Implementación**

Construir un software que funcione con la arquitectura planteada y que satisfaga los requerimientos establecidos.

**e) Pruebas**

Comprobar que el comportamiento establecido en los requerimientos es satisfactorio y tiene todas las características que se solicitaron.

**f) Despliegue**

Realizar distribuciones del software y compartirlo a los usuarios.

**2.1.6. Disciplina de soporte RUP**

Establece la documentación que se necesita para la realización del sistema.

**a) Configuración y administración del cambio**

Almacena la totalidad de versiones del sistema.

**b) Administración del proyecto**

Gestiona el calendario con sus respectivos recursos que se van a utilizar.

**c) Ambiente**

Controla y determina el ambiente de desarrollo del software.

**d) Distribución**

Encargarse de todo lo indispensable para la salida del sistema.

**2.1.7. Elementos de RUP**

**a) Actividades**

Tareas a realizarse en cada iteración.

**b) Trabajadores**

Encargados de cada tarea de una iteración.

**c) Artefactos**

Documentos que se realizan para tener una base con la cual hacer el sistema.

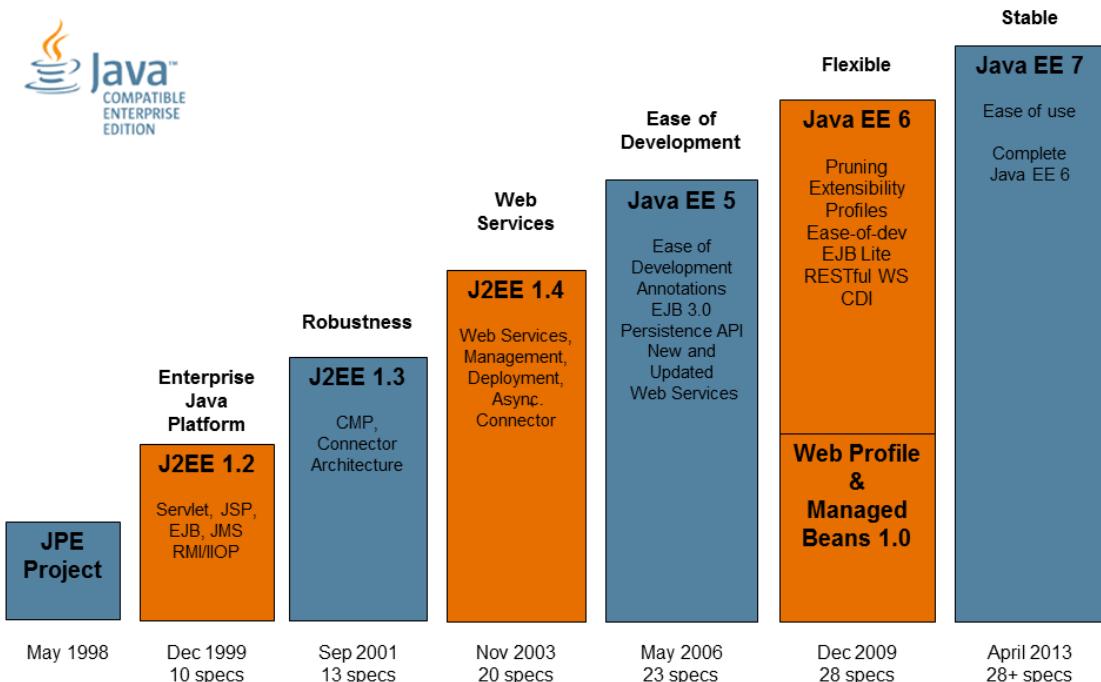
**2.2 JEE**

**2.2.1. Introducción**

Es una plataforma de desarrollo que sirve para programar y ejecutar software de aplicaciones en lenguaje Java y crear aplicaciones empresariales a la medida, ha estado enfocada en brindar transaccionalidad, seguridad, interoperabilidad, persistencia, objetos distribuidos, entre muchos servicios más durante más de una década, por tal motivo actualmente es una de las plataformas más maduras, estables y confiables para el desarrollo de sistemas de información.

Los programadores pueden poner atención en las reglas de negocio en vez de concentrarse en actividades de bajo nivel debido a que la seguridad, escalabilidad,

transacciones, administración de los componentes desplegados y concurrencia pueden ser controlados por el servidor de aplicaciones.



**Figura 8. Historial de las versiones de Java**

Fuente: (Oracle, 2015)

JEE da la posibilidad de utilizar arquitecturas de varias capas distribuidas, utiliza un conjunto de servicios con distintas funcionalidades definidas en módulos que se ejecutan en un servidor de aplicaciones. La comunidad de desarrollo Java Community Process establece especificaciones para Java EE como su arquitectura. Todas las especificaciones que son hechas por esta comunidad son consideradas como un estándar. Los desarrolladores tienen que formalizar varios requerimientos que establece la comunidad para poder constar que sus especificaciones son acordes a JEE.

### 2.2.2. Arquitectura multicapas de JAVA

Un proyecto JEE en Java está compuesto de varias capas, cada una se encarga de una ocupación distinta. Segregar una aplicación en capas tiene diferentes beneficios, como son: autonomía de responsabilidades, se puede actualizar fácilmente, especialización de los programadores en cada capa, etc.



**Figura 9. Arquitectura Multicapas**

Fuente: (Oracle, 2015)

JEE se apoya en diferentes apis en cada capa de un sistema.

#### a) Capa Web:

Es la interacción del cliente y el sistema por medio de un navegador Web o una aplicación de escritorio, se llama también capa Cliente. Cuando el sistema es web, este reside en un servidor web. Java server pages y java server faces son los marcos de trabajo más utilizados en esta capa.

#### b) Capa de Negocio

Compuesta por los Beans de java.

#### c) Capa de Datos

En esta capa se manejan apis como JDBC, o JPA. Estas tecnologías nos van a permitir gestionar cada modelo particular de base de datos, creando una comunicación fluida en la cual se pueda consultar y guardar información desde el sistema en la base de datos.

#### 2.2.3. Contenedores JAVA EE

Los contenedores brindan asistencia a los componentes, sirven para enlazar el código de bajo nivel de las plataformas con los componentes, además ayudan a hacer más simple la construcción de componentes escondiendo partes engorrosas. Tienen una tarea diferente según la capa a la que corresponde.

Los contenedores ofrecen servicios de seguridad, gestión de transacciones, consulta de directorios de nombres (JNDI), etc.



**Figura 10. Contenedores JEE**

Fuente: (Oracle, 2015)

**a) Servidor de Aplicaciones JEE:**

Es el entorno de ejecución de aplicaciones que provee el contenedor Web y/o el contenedor de Enterprise beans.

**b) Contenedor EJB (Enterprise Java Beans):**

Controla la ejecución (ciclo de vida) de los EJBs, da soporte a los componentes que implementan la capa de lógica de negocio, ofrece servicios de control de acceso y seguridad, controla transacciones y ejecución concurrente.

**c) Contenedor Web:**

Trabaja en programas de JEE alojados en un servidor de aplicaciones. Su objetivo es gestionar la ejecución de java server pages, servlets, páginas Web y varios servicios de Enterprise JavaBeans. Ejemplos: Tomcat, Jetty.

**d) Contenedor aplicación cliente:**

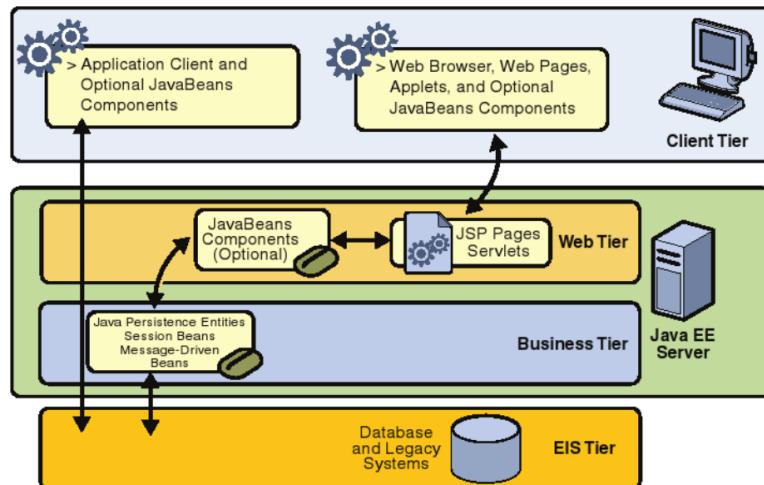
Ofrece la infraestructura necesaria para la ejecución del cliente.

### e) Contenedor de Applets:

Está encargado de gestionar la ejecución de applets, se ejecuta en el computador cliente, sus componentes son un plugin y un navegador.

#### 2.2.4. Componentes JEE

Son objetos java que se ensamblan y ejecutan dentro de un servidor JEE.



**Figura 11. Componentes JEE**

Fuente: (Oracle, 2015)

#### a) Clientes de la aplicación

Son los clientes web y de escritorio como swing y awt.

#### b) Componentes Web

Está conformado por los servlets, páginas JSP y JSF.

#### c) Componentes de negocio

Son los Enterprise Java Beans EJB y las entidades Java Persistence API.

#### d) Capa Enterprise Information Systems EIS

Son las bases de datos o aplicaciones heredadas (legacy systems) que actúan como bodegas de datos.

### 2.2.5. Servicios estándar y tecnologías de soporte JEE

#### a) Java Database Connectivity API (JDBC)

Es usado para enviar comandos SQL hacia una base de datos relacional, que puede ser Oracle, Infomix, SyBase, etc. Establece una conexión con una BD, envía sentencias SQL y procesa los resultados.

#### b) Java Transaction API (JTA) o Java Transaction Service (JTS)

Tecnología que genera una abstracción sobre la gestión de transacciones entre varios sistemas, permitiendo transacciones distribuidas.

#### c) Java Message Service API (JMS)

Es una solución de Java que establece en una plataforma una manera de comunicarse entre aplicaciones fundamentada en el paso de mensajes. Estos textos enviados y recibidos posibilitan a las aplicaciones a percibirse entre ellas y mantener comunicación de manera asíncrona. Los mensajes que están en una cola pueden ser receptados por uno o por diferentes destinatarios dependiendo la necesidad.

#### d) Java Naming and Directory Interface (JNDI)

Es una api que brinda servicios de nombrado y directorio a las soluciones informáticas escritas. Brinda la opción de poder ingresar a diferentes directorios al mismo tiempo debido a que está construido para no depender de ninguna implementación de funcionalidad de directorios.

#### e) JavaMail API

Librería desarrollada para realizar envíos de correos electrónicos directamente desde una aplicación Java

#### f) Java API for XML Processing JAXP

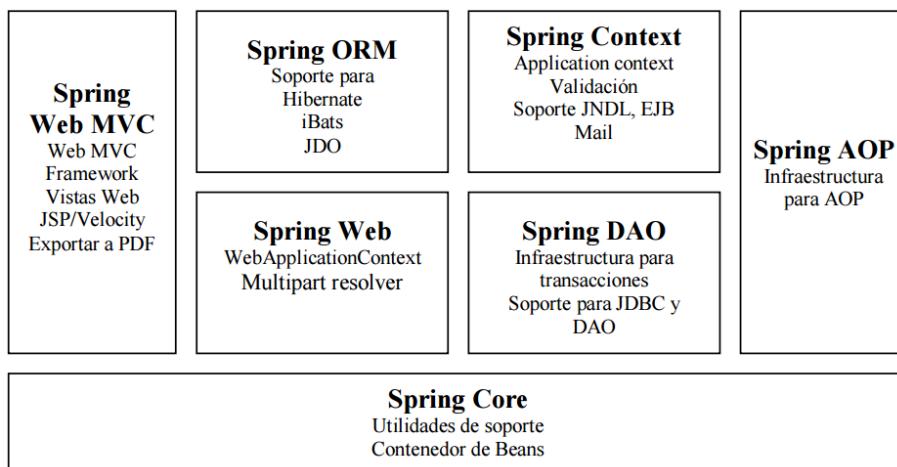
Api que da tratamiento a documentos xml.

### g) Java Persistence API (JPA)

Es una especificación de Java para la persistencia de objetos Java a cualquier base de datos relacional.

## 2.3 FRAMEWORK SPRING

Spring es un marco de trabajo que se divide en módulos, está provisto de una arquitectura segregada en siete capas, como aparece en la figura 12, el desarrollador puede valerse solamente de los módulos que necesita dependiendo del sistema que requiera construir.



**Figura 12. Arquitectura de Spring**

Fuente: (Balani, 2005)

Las siguientes son varias de las propiedades de este marco de trabajo:

- **Contenedor IoC:**

Este se encarga de configurar y gestionar los beans que se alojan en el contenedor. Es el núcleo del framework.

- **Soporte para Programación Orientada a Aspectos AOP:**

Para la implementación de este soporte en aplicaciones Spring brinda una tecnología fundamentada en Proxies, otra manera de establecer este soporte es utilizar AspectJ.

- **Simple Acceso a datos:**

Se puede usar JDBC el cual es muy consistente, otra opción bastante estable y que facilita la gestión de datos es usar uno de los marcos de trabajo con los cuales Spring se

puede fusionar, además de esto brinda su jerarquía de excepciones en las que se toman las excepciones arrojadas por los diferentes marcos de trabajo de gestión de datos, esto faculta cambiar de marco de trabajo en cualquier instante sin que el código sea afectado.

- **Gestión transaccional:**

Este framework consta de abstracciones las cuales brindan la posibilidad de manejar requerimientos transaccionales, permite usar transacciones JDBC, Hibernate, JTA ,etc.

- **Modulo Web:**

El framework proporciona un patrón de arquitectura MVC fundamentado en peticiones.

#### **Beneficios de usar el framework Spring:**

- Mejora el diseño de la programación orientada a objetos.
- Se puede centrar la atención en las reglas del negocio.
- No se necesita generar tanto código para la infraestructura.
- Se facilita la realización de pruebas unitarias
- El servidor de aplicaciones es independiente.
- No se necesita configurar APIs para que funcione.

## **2.4 Maven**

### **2.4.1. Introducción**

Maven es una palabra judía que significa acumulador de conocimientos. Maven se inició originalmente como un intento de simplificar los procesos de construcción de un proyecto llamado Jakarta Turbine.

Hubo varios proyectos, cada uno con su propia generación de archivos que estaban todos un poco diferentes y los JARs fueron chequeados en CVS.

Se quería una forma estándar para construir los proyectos, una definición clara de un proyecto, una manera fácil de publicar información del proyecto y una forma de compartir archivos JAR a través de varios proyectos.

El resultado es una herramienta que ahora se puede utilizar para la construcción y la gestión de cualquier proyecto basado en Java. Los creadores de Maven esperan haber creado algo que hará que el trabajo del día a día de los desarrolladores de Java más fácil y generalmente ayudar en la comprensión de cualquier proyecto basado en Java.

Poder usarse en red es una de las propiedades principales de Maven. Tiene un motor dentro de su kernel, el cual tiene la capacidad de tomar un repositorio para conseguir de estos, los plugins que necesita, con esto da la facilidad de acceder a los proyectos de código abierto en Java, también de Apache y diferentes empresas de desarrollo que se requiera por más que pertenezcan a diferentes versiones. Se ha pretendido calificar cual es el mejor, ese repositorio o al que le sucede osea el repositorio Maven 2 para adoptarlo como el oficial en las distribuciones Java, pero no se ha terminado de definir este dilema.

Este framework a más de ofrecer el servicio de descargar archivos de su base de datos, brinda la posibilidad también de subirlos a la misma base de datos cuando haya finalizado un proyecto, permitiendo usar los nuevos archivos por otras personas. La sincronización de las salidas de archivos la realiza una cache.

La arquitectura de este framework está fundamentada en plugins, esto facilita su utilización a través de un proyecto, manejándola por medio de su entrada estándar.

Está abierta la posibilidad de hacer plugins para realizar una interfaz entre cualquier lenguaje de programación y programas como compiladores, frameworks de pruebas unitarias, etc. Pero la actualmente hay muy poca utilización y mantenimiento de lenguajes diferentes de java.

Hoy por hoy Maven mantiene el plugin para el marco de trabajo .net y anteriormente mantenía el plugin de C y C++.

#### **2.4.2. Objetivos de Maven**

La meta primordial de este framework es que la persona que desarrolla comprenda la total situación del sistema lo más brevemente posible. En la búsqueda de esta meta el framework tiene que lidiar con varios aspectos como son:

- Mantener sencillo el conjunto de actividades del desarrollo
- Establecer un método para construir con características similares
- Brindar datos seguros y precisos
- Brindar directrices para mejorar los procedimientos
- Facilitar las actualizaciones

#### **Mantener sencillo el conjunto de actividades del desarrollo**

Mediante el uso de Maven no se necesita saber sobre el código de bajo nivel, el framework brinda grandes cantidades de blindaje para los detalles.

## **Establecer un método para construir con características similares**

Maven permite a un proyecto ser construido usando su modelo de objeto de proyecto (POM) y un varios plugins que son usados por todos los proyectos, brindando un sistema de construcción uniforme. Una vez se está familiarizado con cómo se construye un proyecto Maven, se sabe automáticamente como se ha construido cualquier proyecto Maven, esto permite ahorrar enormes cantidades de tiempo cuando se trata de navegar muchos proyectos.

## **Brindar datos seguros y precisos**

Maven proporciona un montón de información útil sobre el proyecto que es, en parte, tomado de su POM y en parte generada a partir de recursos de su proyecto. Por ejemplo, Maven puede proporcionar:

- Documento de registro de cambios creado directamente de control de código fuente
- Recursos de referencias cruzadas
- Listas de correo
- Lista de Dependencia
- Informes de pruebas unitarias incluyendo cobertura

Al tiempo que Maven mejora, el conjunto de información proporcionada mejorará, todo lo cual será transparente para los usuarios de Maven.

Otros productos pueden también proporcionar plugins de Maven para permitir que su sistema de información del proyecto se junte con parte de la información estándar dada por Maven, todo esto sigue basándose en el POM.

## **Brindar directrices para mejorar los procedimientos**

Maven tiene como objetivo reunir a los principios actuales para mejorar el procedimiento de desarrollo, y hacerlo facil para guiar un proyecto en esa dirección.

Por ejemplo, la especificación, ejecución y presentación de informes de las pruebas unitarias son parte del ciclo de construcción normal utilizando Maven. Las actuales mejores prácticas de pruebas unitarias se utilizaron como pautas:

- Mantener las pruebas de código fuente en un separado pero paralelo árbol de recursos.
- El uso de convenciones de nombres de casos de prueba para localizar y ejecutar pruebas.

- Tener casos de prueba configurando su entorno y no se basan en la personalización de la construcción para la preparación de la prueba.

Maven también tiene como objetivo asistir en el flujo de trabajo del proyecto, tales como la gestión de la liberación y seguimiento de problemas.

Maven también sugiere algunas pautas sobre cómo diseñar la estructura de directorios de su proyecto para que una vez que aprenda el diseño, pueda navegar con facilidad cualquier otro proyecto que utiliza Maven y también poder tener los mismos valores predeterminados.

### **Facilitar las actualizaciones**

Maven proporciona una manera fácil a sus clientes para actualizar sus instalaciones, para que puedan tomar ventaja de los cambios que se han hecho a Maven.

#### **2.4.3. Pom**

POM significa "Modelo de objetos del proyecto". Es una representación XML de un proyecto Maven que se encuentra en un archivo llamado pom.xml. Cuando se habla de un proyecto maven se está hablando en el sentido filosófico, más allá de una mera colección de archivos que contienen código. Un proyecto maven contiene los archivos de configuración, así como los desarrolladores involucrados y las funciones que desempeñan, el sistema de seguimiento de defectos, la organización y licencias, la URL en donde el proyecto vive, dependencias del proyecto, y todas las otras piezas pequeñas que entran a jugar para dar vida al código. Es una ventanilla única para todo lo relacionado con el proyecto. De hecho, en el mundo Maven, un proyecto no contiene ningún código en absoluto, simplemente un pom.xml.

En la figura 13 se muestra una lista de los elementos directamente bajo el elemento del proyecto del POM. Observe que la versión de modelo que contiene 4.0.0, es actualmente la única versión de POM soportada tanto para Maven 2 y 3, y siempre es requerida.

```
1. <project xmlns="http://maven.apache.org/POM/4.0.0"
2.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
4.                     http://maven.apache.org/xsd/maven-4.0.0.xsd">
5.   <modelVersion>4.0.0</modelVersion>
6.
7.   <!-- The Basics -->
8.   <groupId>...</groupId>
9.   <artifactId>...</artifactId>
10.  <version>...</version>
11.  <packaging>...</packaging>
12.  <dependencies>...</dependencies>
13.  <parent>...</parent>
14.  <dependencyManagement>...</dependencyManagement>
15.  <modules>...</modules>
16.  <properties>...</properties>
17.
18.  <!-- Build Settings -->
19.  <build>...</build>
20.  <reporting>...</reporting>
21.
22.  <!-- More Project Information -->
23.  <name>...</name>
24.  <description>...</description>
25.  <url>...</url>
26.  <inceptionYear>...</inceptionYear>
27.  <licenses>...</licenses>
28.  <organization>...</organization>
29.  <developers>...</developers>
30.  <contributors>...</contributors>
31.
32.  <!-- Environment Settings -->
33.  <issueManagement>...</issueManagement>
34.  <ciManagement>...</ciManagement>
35.  <mailingLists>...</mailingLists>
36.  <scm>...</scm>
37.  <prerequisites>...</prerequisites>
38.  <repositories>...</repositories>
39.  <pluginRepositories>...</pluginRepositories>
40.  <distributionManagement>...</distributionManagement>
41.  <profiles>...</profiles>
42. </project>
```

**Figura 13. Elementos del POM**

Fuente: (Maven, 2016)

El POM es un archivo de configuración el cual contiene la información total que necesita un proyecto, también contiene las configuraciones de plugins a utilizarse mientras dure el proceso de desarrollo. Esto viene a responder las preguntas "quién", "qué" y "dónde", en cambio el ciclo de vida de construcción es el "cuándo" y "cómo". A pesar de que pareciera que el POM no podría modificar el flujo del ciclo de vida, este si tiene esa capacidad. Por ejemplo, al configurar el maven-antRun-plugin, se podría integrar de forma efectiva las tareas ant dentro del POM. A final de cuentas es una declaración. A pesar de eso si es que un build.xml le ordena exactamente a un ant qué hacer cuando se ejecuta (procedural), un POM fija su estructura (declarativa). Si por algún motivo externo el ciclo de vida omite la ejecución del plugin de ant, esto no va a parar a los plugins que se están ejecutando. Esto difiere de un archivo build.xml, donde las labores casi siempre dependen de las líneas que se ejecutaron anteriormente.

```

1. <project xmlns="http://maven.apache.org/POM/4.0.0"
2.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
4.                     http://maven.apache.org/xsd/maven-4.0.0.xsd">
5.   <modelVersion>4.0.0</modelVersion>
6.
7.   <groupId>org.codehaus.mojo</groupId>
8.   <artifactId>my-project</artifactId>
9.   <version>1.0</version>
10.  </project>
```

**Figura 14. Ejemplo pom.xml**

Fuente: ([Maven, 2016](#))

#### 2.4.4. Coordenadas de maven

El POM definido anteriormente es el mínimo que tanto Maven 2 y 3 permitirá. groupId:artifactId:version son todos los campos requeridos (aunque, groupId y version no es necesario definir explícitamente si se heredan de un parent). Los tres campos actúan muy parecido a una dirección, fecha y hora en uno solo. Esto marca un lugar específico en un repositorio, que actúa como un sistema de coordenadas para proyectos Maven.

**a) groupId:**

Esto generalmente es único a lo largo de la organización o del proyecto. Por ejemplo, todos los artefactos del núcleo de Maven hacen (bueno, deberían) vivir bajo el org.apache.maven del groupId. El ID de grupo no utilizan necesariamente la notación de puntos, por ejemplo, el proyecto junit. Se debe tener en cuenta que el groupId con notación de puntos no tiene que corresponder a la estructura de paquete que contiene el proyecto. Eso es, sin embargo, una buena práctica a seguir. Cuando se almacena en un repositorio, el grupo actúa como la estructura de paquetes de Java en un sistema operativo. Los puntos son reemplazados por separadores de directorio específico del sistema operativo (como '/' en Unix) que se convierte en una estructura de directorios relativa de un repositorio base. Ejemplo: en la Figura 14 el grupo org.codehaus.mojo vive dentro del directorio \$ M2\_REPO / org / Codehaus / mojo.

**b) artifactId:**

El artifactId generalmente es el nombre por el cual el proyecto es conocido. Aunque el groupId es importante, la gente dentro del grupo rara vez mencionan al groupId en la discusión (que a menudo todos tienen el mismo ID, como el proyecto Codehaus Mojo groupId: org.codehaus.mojo). A lo largo del proyecto se crea una llave con el groupId, que separa a este proyecto de cualquier otro proyecto en el mundo. Junto con el groupId, el artifactId define totalmente la donde vive del artefacto dentro del repositorio. En el caso del proyecto anterior, my-project vive en \$ M2\_REPO /org/Codehaus/mojo/my-project.

**c) version:**

Esta es la parte faltante en lo que se refiere a nombres. groupId: artifactId son un solo proyecto, pero estos no pueden definir de que tipo proyecto estamos hablando. Ejemplo: para diferencias el junit: junit de la actual versión que es la numero 4, o el de hace cuatro años que es la versión 2 En resumen: los cambios de código, esos cambios deben ser versionados, y este elemento mantiene esas versiones en la fila. Esto también se utiliza en el repositorio de un artefacto para separar las versiones de una a otra. Los archivos de my-project de la versión 1.0 viven en la estructura de directorios \$ M2\_REPO / org / Codehaus / mojo / my-project / 1.0 my-project.

Los tres elementos mencionados anteriormente apuntan a una versión específica de un proyecto, de esta manera permiten a Maven saber con quién estamos tratando, y en que parte de su ciclo de vida del software los queremos.

#### d) packaging:

Ahora que se tiene la estructura de dirección del groupId: artifactId:version, hay otra etiqueta estándar tener una dirección realmente completa. Ese es el tipo de artefacto del proyecto. En nuestro caso, el ejemplo POM para org.codehaus.mojo: my-project: 1.0 definido anteriormente se empaquetaría como un jar. Se podría convertirlo en un war al declarar un empaquetado distinto:

```

1. <project xmlns="http://maven.apache.org/POM/4.0.0"
2.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
4.                     http://maven.apache.org/xsd/maven-4.0.0.xsd">
5. ...
6. <packaging>war</packaging>
7. ...
8. </project>
```

**Figura 15. Empaquetado war**

**Fuente: (Maven, 2016)**

Cuando se declara sin empaquetado, Maven admite el artefacto como el jar predeterminado. Los tipos válidos son role-hints de Plexus del rol de componentes org.apache.maven.lifecycle.mapping.LifecycleMapping. Los valores centrales de empaquetado actuales son: pom, jar, maven-plugin, ejb, war, ear, rar, par. Estos establecen la lista por defecto de los objetivos que se ejecutan para cada fase del ciclo de vida de construcción correspondiente para una estructura de paquete en particular. A veces se verá que Maven imprime una de las coordenadas del proyecto como groupId:artifactId:packaging:version.

#### e) classifier:

Usted puede encontrar ocasionalmente un quinto elemento en las coordenadas, eso es el classifier. Este tipo de proyectos se muestran como groupId:artifactId:packaging:classifier:version.

#### 2.4.5. Relaciones del pom

Un aspecto de gran alcance de Maven es el manejo de las relaciones de los proyectos; que incluye dependencias (y dependencias transitivas), la herencia y agregación (proyectos multi-módulo). La gestión de la dependencia tiene una larga tradición de ser

complicado para cualquier cosa incluyendo al más trivial de los proyectos. "Jarmageddon" sobreviene rápidamente como un ejemplo de que el árbol de dependencia se hace grande y complicado. "Jar Hell" sigue, donde las versiones de dependencias en un sistema no son equivalentes a las versiones con las cuales fueron desarrolladas, ya sea por la versión incorrecta dada, o versiones contradictorias entre jars con nombres similares. Maven resuelve ambos problemas a través de un repositorio local común desde el cual enlaza los proyectos correctamente, versiones y todo.

## Dependencias

La piedra angular del POM es su lista de dependencias. La mayoría de los proyecto depende de los demás para ser construido y funcionar correctamente, Maven gestiona esta lista para el desarrollador, con esto se ha ganado mucho. Maven descarga y enlaza las dependencias en la compilación y otros objetivos que los requieran. Como bono adicional, Maven trae las dependencias de esas dependencias (dependencias transitivas), lo que permite a su lista centrarse exclusivamente en las dependencias que el proyecto requiere.

```

1. <project xmlns="http://maven.apache.org/POM/4.0.0"
2.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
4.                     https://maven.apache.org/xsd/maven-4.0.0.xsd">
5. ...
6. <dependencies>
7.   <dependency>
8.     <groupId>junit</groupId>
9.     <artifactId>junit</artifactId>
10.    <version>4.0</version>
11.    <type>jar</type>
12.    <scope>test</scope>
13.    <optional>true</optional>
14.  </dependency>
15. ...
16. </dependencies>
17. ...
18. </project>
```

**Figura 16. Dependencias del POM**

Fuente: (Maven, 2016)

- **groupId, artifactId, version:**

Estos elementos se ven muy a menudo. Esta trinidad se utiliza para calcular las coordenadas de Maven de un proyecto específico en el tiempo, demarcándolo como una dependencia de este proyecto. El propósito de este cálculo es seleccionar una versión que coincide con todas las declaraciones de dependencia (debido a las dependencias transitivas, puede haber múltiples declaraciones de dependencias para el mismo artefacto). Los valores deben ser: **groupId, artifactId:** directamente las coordenadas correspondientes de la dependencia.

#### **Version:**

Una especificación de requisitos para la versión de dependencia, que se utilizará para calcular la versión efectiva de la dependencia.

Desde que la dependencia es descrita por las coordenadas de Maven.

Esto obliga a depender únicamente de dependencias que Maven puede gestionar. Hay veces, por desgracia, cuando un proyecto no puede ser descargado desde el repositorio central de Maven. Por ejemplo, un proyecto puede depender de un jar que tiene una licencia de código cerrado que le impide estar en un repositorio central. Hay tres métodos para hacer frente a este escenario.

- i. Instalar la dependencia local usando el plugin de instalar. Este método es el método más simple recomendado.

```
mvn install:install-file -Dfile=non-maven-proj.jar -DgroupId=some.group -DartifactId=non-maven-proj -Dversion=1 -Dpackaging=jar
```

**Figura 17. Comando de instalación individual de dependencia**

**Fuente: (Maven, 2016)**

Nótese que aún se requiere una dirección, sólo que esta vez se utiliza la línea de comandos y el plugin de instalación creará un POM para usted con la dirección indicada.

- ii. Crear un repositorio propio y desplegarlo allí. Este es un método preferido para las empresas con intranet y la necesidad de ser capaz de mantener a todos en sincronía. Hay una meta de Maven llamado

deploy:deploy-file la cual tiene la misma función de install:install-archivo.

- iii. Establecer el ámbito de la dependencia del sistema y definir un systemPath.

- **classifier:**

El classifier da la opción de como su nombre lo indica clasificar los artefactos que tienen diferente contenido pero se encuentran en el mismo POM. Es una adición opcional y arbitraria que si se presenta es añadida al nombre del artefacto después del número de versión.

Un caso que se da muy a menudo y que es necesario tomarlo en consideración es, por ejemplo, un proyecto que presenta un artefacto dirigido JRE 1.5, pero al mismo tiempo también otro artefacto que todavía apoya JRE 1.4. El primer artefacto podría estar equipado con el classifier jdk15 y el segundo con jdk14 de tal manera que los clientes pueden elegir cuál usar.

Otro caso de uso común para los classifier es la necesidad de colocar los artefactos secundarios al artefacto principal del proyecto. Si se explora el repositorio central de Maven, se puede notar que los sources y javadocs de classifier se utilizan para implementar el código fuente del proyecto y la documentación de la API, junto con los archivos de clase empaquetados.

- **type (tipo):**

Corresponde al dependiente tipo de packaging del artefacto. Esto da el JAR por defecto. Si bien por lo general representa la extensión del nombre de archivo de la dependencia, que no siempre es el caso. Un tipo puede ser mapeado a una extensión diferente y un clasificador. El tipo a menudo se corresponde con el empaquetado utilizado, aunque esto también no es siempre el caso. Algunos ejemplos son jar, ejb-client y test-jar. Nuevos tipos se pueden definir por los plugins que establecen extensions a true, así que esto no es una lista completa.

- **scope (alcance):**

Este elemento se refiere a la ruta de clase de la tarea en cuestión (compilación y tiempo de ejecución, pruebas, etc.), así como la forma de limitar la transitividad de una dependencia. Hay cinco ámbitos disponibles:

- **Compile**

Este es el ámbito predeterminado, utilizado si no se especifica ninguno. Su objetivo es compilar dependencias que estén desocupadas en cada ruta de clases. Después de eso, esas dependencias se difunden a los proyectos que dependen de estas.

- **Provided**

Este es muy parecido a compile, pero con la diferencia que espera el JDK o un contenedor para proporcionar en tiempo de ejecución. Este sólo se lo puede utilizar en la compilación y prueba de ruta de clases, y no es transitivo.

- **Runtime**

Este ámbito hace ver que la dependencia no se necesita para compilación, sino que es requerida para su ejecución. Este está en las rutas de clases de tiempo de ejecución y de prueba, pero no en la ruta de clases de compilación.

- **Test**

El objetivo de este ámbito es mostrar que la dependencia solo está libre para ocuparse en las etapas de compilación y pruebas y no es requerida para el uso normal de la aplicación, además indica que la dependencia es transitiva.

- **System**

Este ámbito es similar a provided, salvo que se tiene que proporcionar el JAR que contiene eso explícitamente. El artefacto está siempre disponible y no se busca en un repositorio.

- **systemPath:**

Se utiliza sólo si el scope de la dependencia es system. De lo contrario, la construcción fallará si este elemento es establecido. La ruta debe ser absoluta, por lo que se recomienda utilizar una propiedad para especificar la ruta específica de la máquina, como \${java.home} / lib. Como se supone que las dependencias de alcance del sistema se

instalan a priori, Maven no comprobará los repositorios para el proyecto, sino que comprueba para asegurarse de que el archivo existe. Si no, Maven fallará la construcción y sugerirá descargar e instalar de forma manual.

- **optional:**

Se marca optional a una dependencia cuando este proyecto en sí es una dependencia. Por ejemplo, Un proyecto A que depende de un proyecto B para compilar una porción de código que no se puede utilizar en tiempo de ejecución, entonces no se necesita del proyecto B para todos los proyectos. Así que si el proyecto X añade al proyecto A como su propia dependencia, a continuación, Maven no necesitará instalar el proyecto B en absoluto. Simbólicamente, si  $=>$  representa una dependencia requerida,  $->$  representa opcional, aunque  $A => B$  puede ser el caso en la construcción de A,  $X => A -> B$  sería el caso en la construcción de X.

En términos simples, cuando se utiliza determinado proyecto, optional permite a otros proyectos saber que dependencias no se necesita para funcionar correctamente.

## **2.5 Tipos de inventarios**

### **2.5.1. Según su función:**

#### **a) Inventario de seguridad o de reserva:**

Como su nombre lo indica este inventario se lo almacena para usarlo si es que ocurre un incidente con la producción y se detiene. También ayuda si es que aumenta la demanda inesperadamente.

#### **b) Inventario de desacoplamiento:**

Se da en el caso de que dos procesos que se relacionan el uno al otro, estos procesos producen o necesitan diferente cantidad de artículos, este inventario almacena los artículos del un proceso hasta que el siguiente los necesite.

**c) Inventario en tránsito:**

En este inventario están los artículos que están siendo utilizados en las actividades distintivas que posibilitan obtener una ventaja competitiva. Estos materiales están aumentando su valor. Son artículos que todavía no se han recibido a pesar de ya haberse pedido.

**d) Inventario de ciclo:**

Es el inventario de artículos acumulados debido a que se compra o se produce más de lo que la empresa necesita, esto se hace con el objetivo de comprar las unidades más baratas porque se adquiere más, o de aumentar la capacidad de la producción.

**e) Inventario de previsión o estacional:**

Constituyen las unidades que se acumulan porque la empresa produjo más artículos de los que necesitaba ese instante, esto ocurre en las épocas de baja demanda con el fin de satisfacer las de alta demanda. Frecuentemente este tipo de inventario se almacena en época de demanda estacional.

**2.5.2. Desde el punto de vista logístico:**

**a) Existencias para especulación:**

Son inventarios almacenados con el fin de especular con el precio así como para satisfacer los requerimientos de la empresa, también pueden ser utilizados para anticiparse a las ventas estacionales o de temporada.

**b) Existencias de naturaleza regular o cíclica:**

Es las unidades almacenadas con el fin de cubrir la demanda promedio hasta que se pueda reaprovisionar el inventario.

**c) Existencias de seguridad:**

Este es creado con el fin de no quedarse sin existencias debido al crecimiento de la demanda, tratando de tener siempre unidades hasta poder reaprovisionarse nuevamente.

**d) Existencias obsoletas, muertas o perdidas:**

Las unidades que permanecen en el inventario cuantioso tiempo normalmente sufren deterioro, caducidad, pérdida o robo.

## CAPÍTULO 3

### FASE ANÁLISIS Y DISEÑO

#### **3.1 Especificación de requerimientos de software**

##### **3.1.1 Introducción**

La especificación de requerimientos de software tiene como propósito detallar los requerimientos funcionales y no funcionales del software, permitiéndome tener una guía para la construcción del sistema a desarrollarse.

Este capítulo describe totalmente el comportamiento externo que tendrá el sistema, enfocado a representar las necesidades del cliente. Se detallara también las restricciones de diseño y otros factores necesarios para proveer una descripción completa y comprensible de los requerimientos de software.

##### **3.1.2 Alcance y definiciones**

Este documento describe de manera detallada los requerimientos del sistema de inventario de elementos para el INAMHI. El sistema tiene como objetivo optimizar el proceso de gestión de elementos, por medio de los módulos que se describen a continuación:

- Gestión de usuarios
- Gestión de elementos
- Gestión de Solicitudes de elementos
- Gestión de Órdenes de trabajo

A continuación las definiciones, acrónimos y abreviaciones a utilizarse en la especificación de requerimientos:

##### **i. De la Aplicación web**

1. Pruebas: Proceso mediante el cual se realizan actividades para verificar la óptima función de la aplicación web.

##### **ii. De tecnología**

1. Sistema de gestión de Base de Datos: Son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

2. Base de Datos: Es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso.
3. Aplicación: Programa informático diseñado para facilitar al usuario la realización de un determinado tipo de trabajo.
4. MySql: Sistema de gestión de base de datos relacional.
5. JEE: Java Enterprise Edition.
6. RUP: Rational Unified Process, metodología estándar para el análisis, diseño, implementación y documentación de sistemas orientados a objetos.
7. JVM: Java Virtual Machine, máquina virtual de java
8. Swing: Gestor de Interfaz Gráfica para Java
9. GUI: Unidad de Interfaz Gráfica.

### **3.1.3 Descripción general**

Los aspectos del levantamiento de requerimientos son esenciales para el desarrollo del aplicativo web, siendo los siguientes:

#### **i. Perspectiva del Producto**

El sistema que se planea construir está enfocado en optimizar el proceso de gestión de elementos del INAMHI, por medio de la implementación de la base de datos que aportará el conocimiento de la existencia, localización y disponibilidad de los elementos y definirá el proceso de asignación de los mismos.

El proyecto constará de varias páginas web que ingresarán detalles de elementos, órdenes de trabajo, reservas, etc. Las cuales por medio de clases en lenguaje java insertarán dicha información en una base de datos relacional especialmente diseñada para que tenga eficiencia y que se pueda seguir utilizando por largo del tiempo.

#### **ii. Funciones del Producto**

El sistema de inventario de elementos para el INAMHI está enmarcado para cumplir con las siguientes funciones:

- Ingreso, consulta, actualización, puesta en marcha, cancelación y cierre de órdenes de trabajo.
- Ingreso, consulta, actualización y eliminación de elementos.
- Ingresar, aprobar, negar y aprobar parcialmente solicitudes de elementos.

### iii. Características del Usuario

Los usuarios son todas aquellas personas involucradas directamente en el uso del sistema.

**Tabla 4.**

#### Usuarios del sistema

Nombre	Descripción
Solicitante OMRA	Hace referencia a los ingenieros asignados a las comisiones. Es el encargado de solicitar el elemento para que se haga una reserva del mismo.
Administrador de Bodega	<p>Este tipo de usuario es quien administra el sistema por lo tanto es considerado como el principal porque hace uso de la mayoría de las funciones del sistema como son:</p> <ul style="list-style-type: none"> <li>○ Gestionar los usuarios.</li> <li>○ Gestionar las órdenes de trabajo de elementos.</li> <li>○ Gestionar elementos.</li> <li>○ Gestionar las solicitudes de elementos.</li> </ul>
Mecánica	Este usuario es quien realiza las reparaciones mecánicas de los elementos, por lo tanto gestionara las órdenes de trabajo de los elementos.
Electrónica	Este usuario es quien realiza las reparaciones electrónicas de los elementos, por lo tanto gestionara las órdenes de trabajo de los elementos.
Metrología	Este usuario es quien calibra los elementos, por lo tanto gestionara las órdenes de trabajo de los elementos.

### iv. Restricciones

- El sistema web no incluirá ningún otro procedimiento de la empresa que no tenga que ver con inventarios de elementos.
- El sistema web no tendrá funcionalidades de configuración.
- La implementación del sistema web se realizará basándose en la metodología RUP para la elaboración del mismo, tomando de ella aquellas características que permitan desarrollar el proyecto de la mejor manera y que se ajusten a las condiciones y tamaño del mismo.
- La implementación del sistema se realizara en un servidor del INAMHI.
- El proyecto tendrá una duración aproximada de 6 meses a partir de la asignación de Director de Tesis.

## v. Suposiciones y dependencias

- Los usuarios finales saben usar un navegador web.
- El director de tesis asignado cuenta con el tiempo necesario para realizar su trabajo.
- INAMHI pondrá a disposición del tesista el servidor y los equipos necesarios para la implementación del sistema.
- INAMHI no realizará cambios significativos en los requerimientos.

### 3.1.4 Requerimientos específicos

Los requerimientos son esenciales, siendo inaceptable que la aplicación web no satisfaga alguno de los mismos. Éstos se han especificado teniendo en cuenta, entre otros, el criterio de testabilidad.

A continuación se detallan los requisitos de cada una de las interfaces externas:

#### i. Interfaz con el usuario

El sistema web tiene que ser amigable y predictivo con el usuario para poder simplificar el trabajo a los ingenieros del INAMHI.

#### ii. Interfaz con el hardware

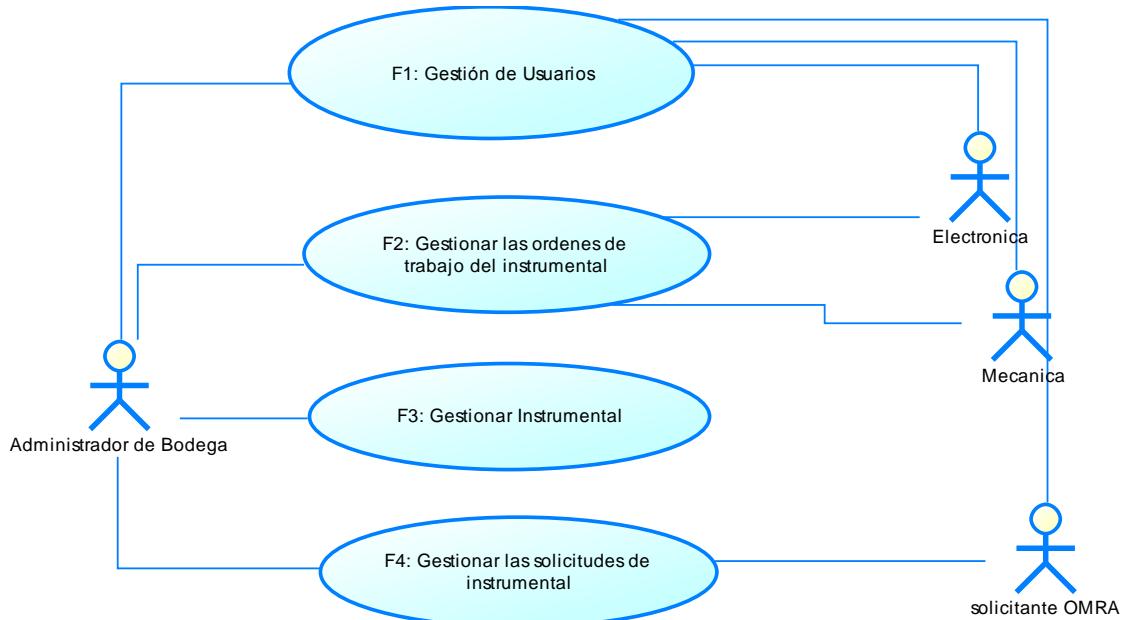
En el manejo del sistema web se puede interactuar con periféricos tanto de entrada, como de salida: mouse, teclado, monitor, pantallas táctiles, botones de navegación, etc.; además de tarjeta de red, puerto de red, etc., que establezcan una conexión a Internet.

#### iii. Interfaz de comunicaciones

Para ingresar al sistema web es indispensable tener acceso a internet en el dispositivo móvil, tablet o computador que se esté utilizando.

### 3.1.5 Requerimientos funcionales

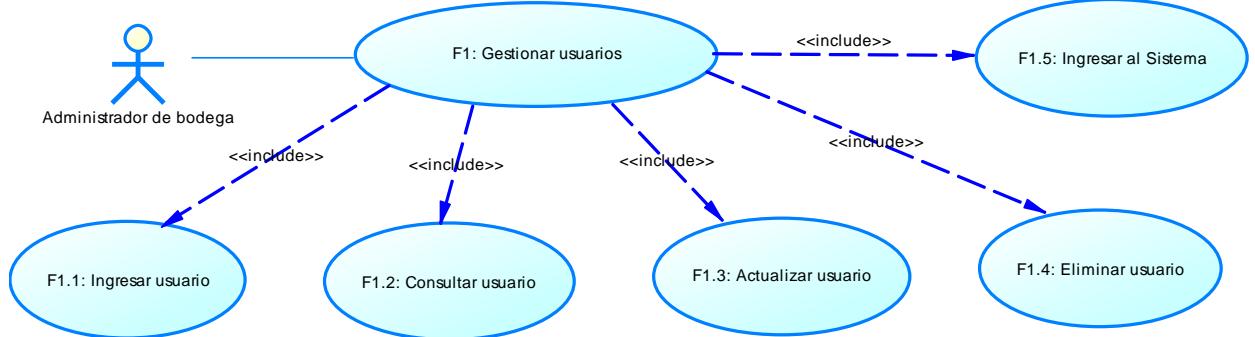
#### i. Diagrama General



**Figura 18. Diagrama General de casos de uso**

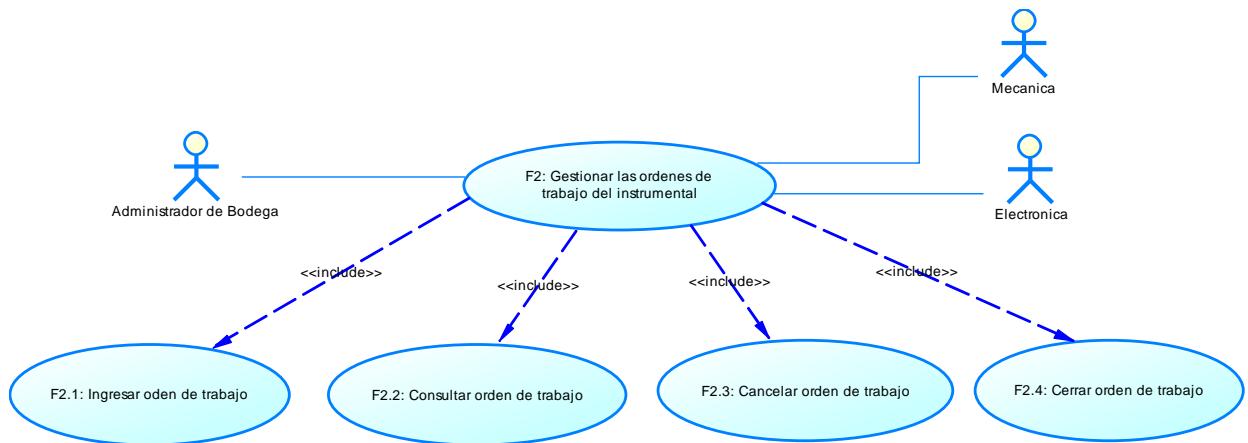
#### ii. Diagramas de Casos de Uso

F1: Gestionar usuarios:



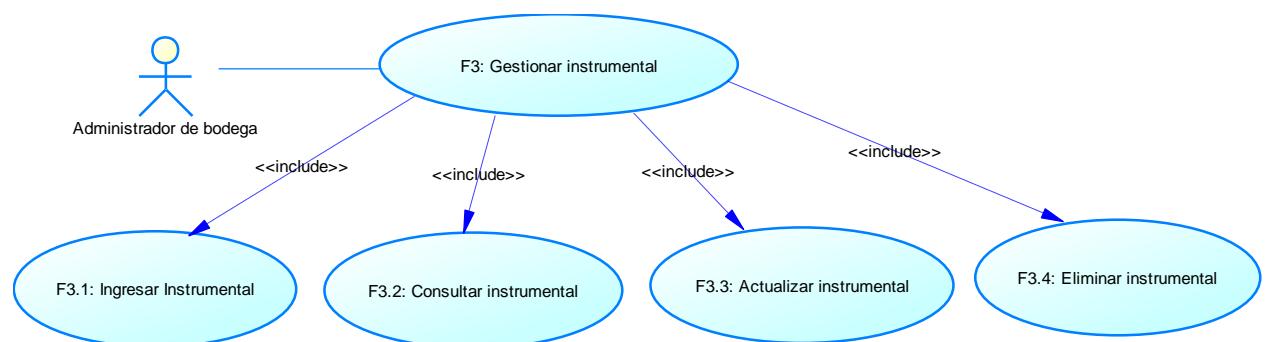
**Figura 19. Módulo Gestionar Usuarios**

## F2: Gestionar las órdenes de trabajo de elementos



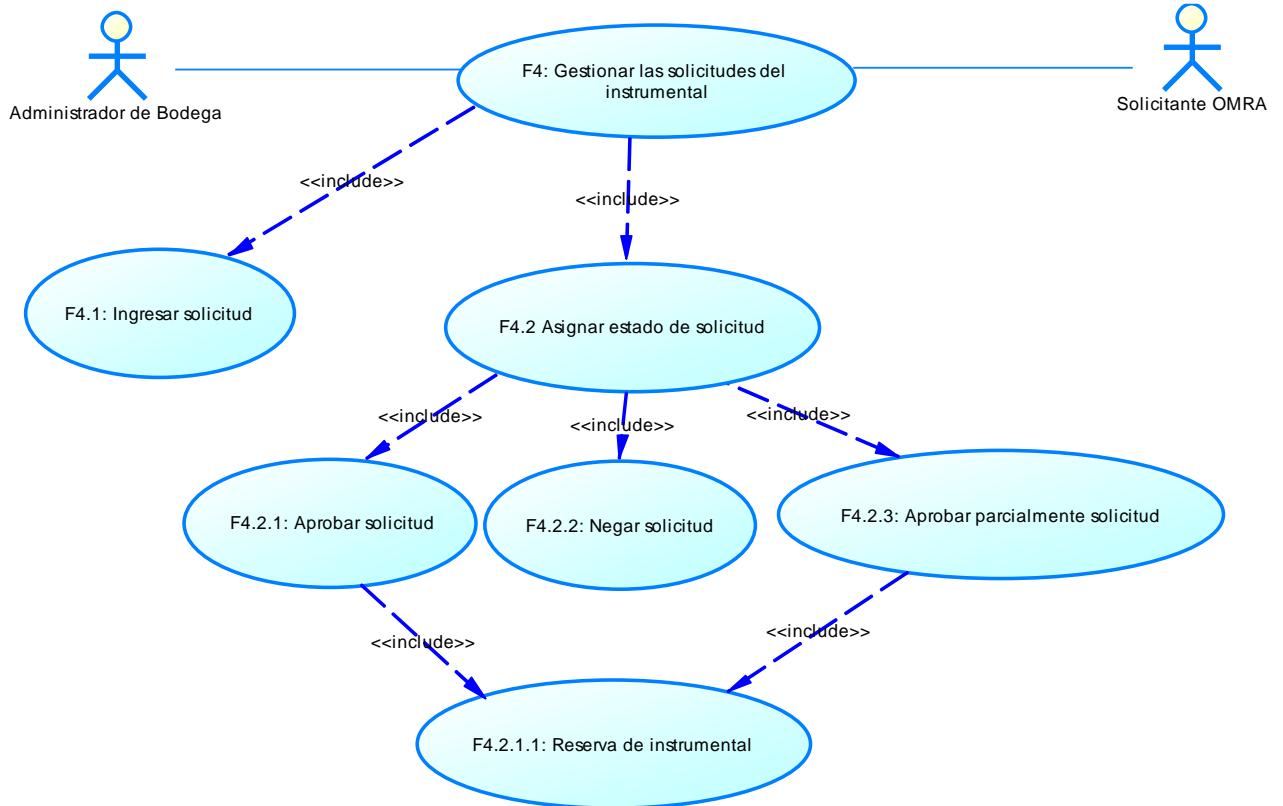
**Figura 20. Módulo Gestionar las órdenes de trabajo de elementos**

## F3: Gestionar elementos



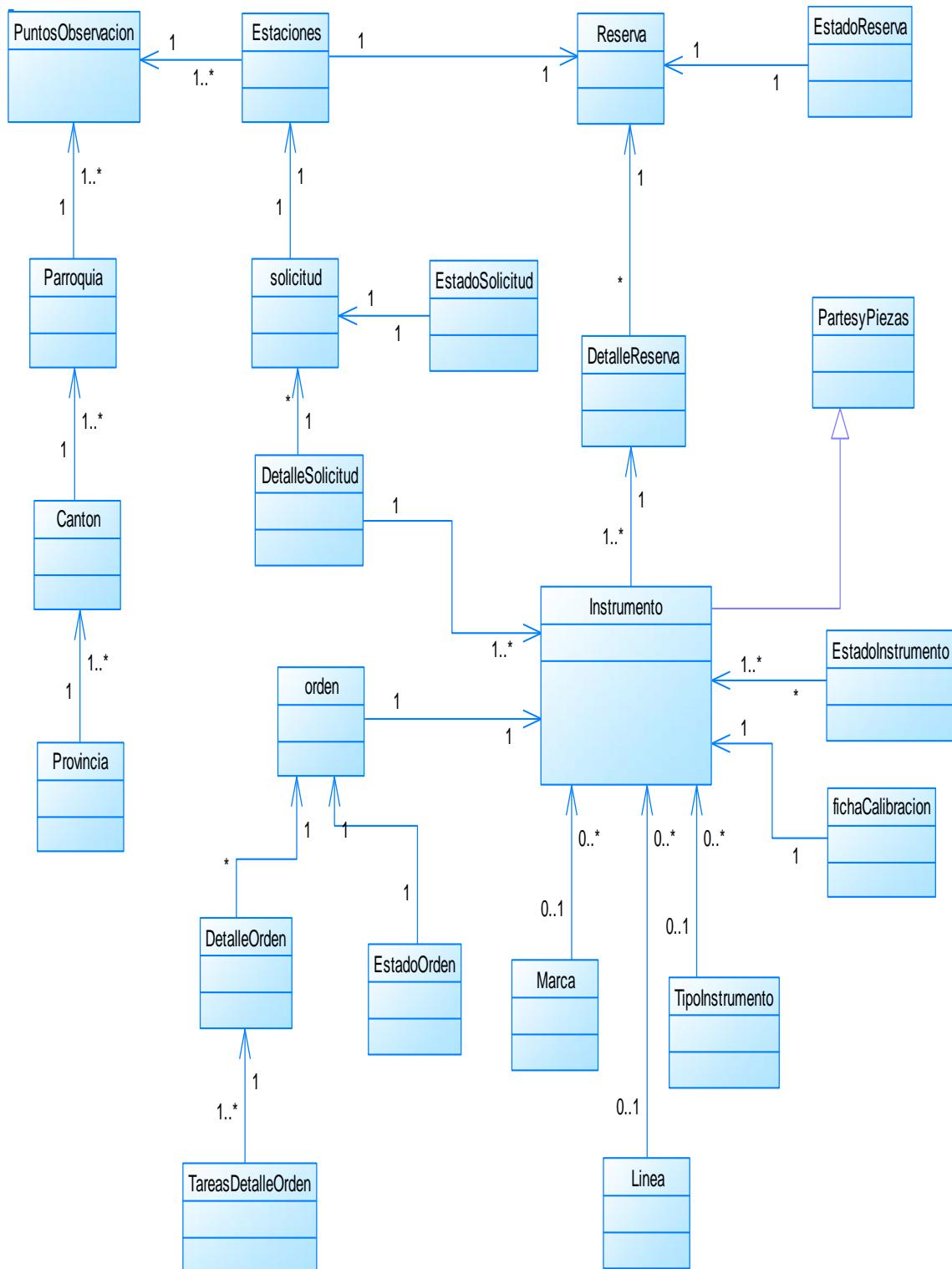
**Figura 21. Gestionar elementos**

#### F4: Gestionar las solicitudes de elementos



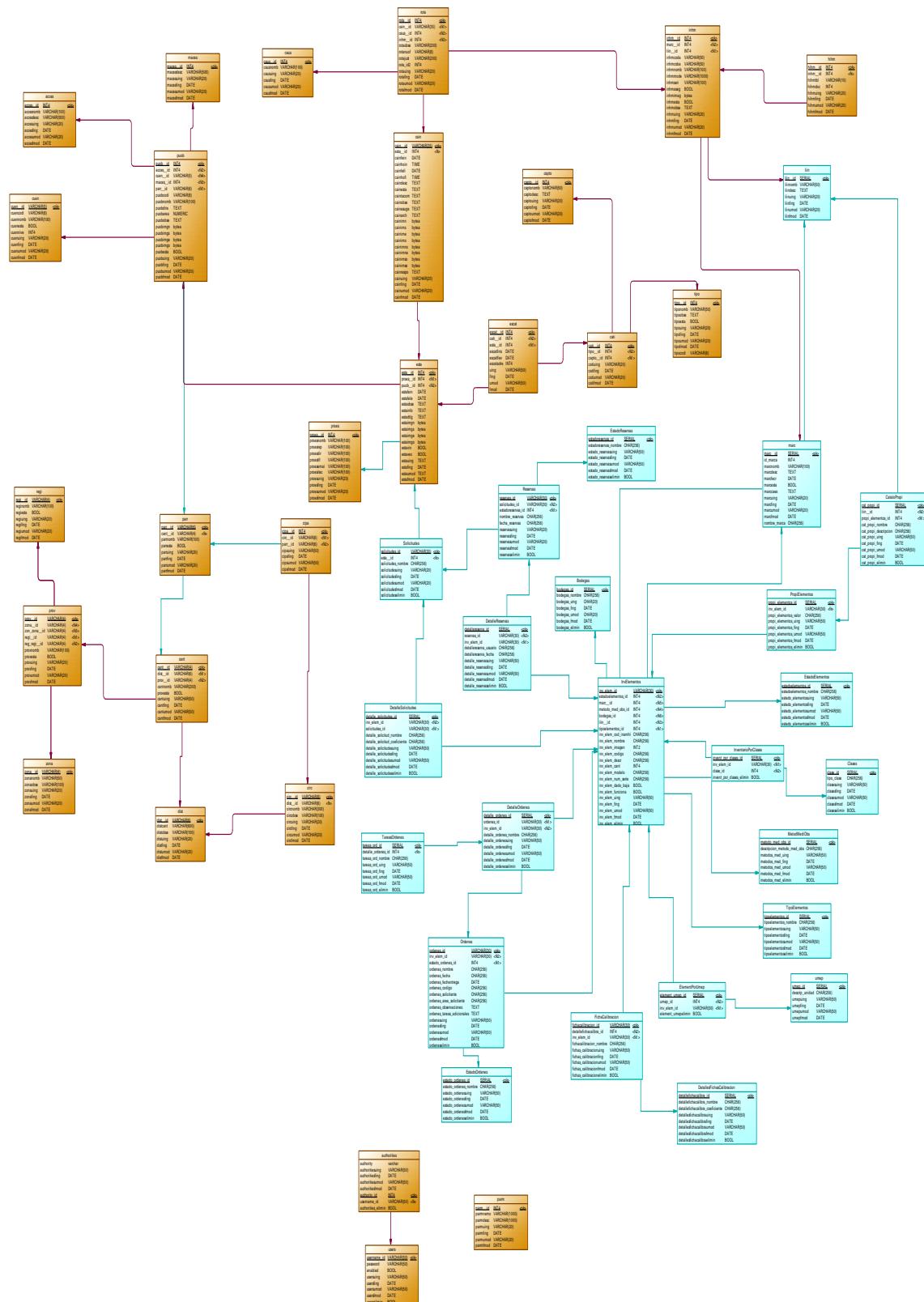
**Figura 22. Gestionar las solicitudes de elementos**

### iii. Diagrama de Clases



**Figura 23. Diagrama de clases**

#### **iv. Modelo Físico de la Base de Datos**



**Figura 24. Diagrama físico**

### **3.1.6 Requerimientos no funcionales**

Los requerimientos no funcionales del presente proyecto son los siguientes:

#### **a) Requisitos de Interfaz**

- La aplicación web tendrá la interfaz en idioma español.
- La aplicación web deberá procurar evitar tipografía compleja, puesto que algunos dispositivos móviles tienen un número reducido de fuentes, además de un soporte limitado para tamaños de fuente y efectos tipográficos (negrita, cursiva, etc.).

#### **b) Requisitos de Navegación**

- El sistema web tiene que utilizar formas consistentes y coherentes de navegación, de esta forma mejora la usabilidad del sistema web, al facultar a los usuarios determinar de forma fácil los patrones de navegación y así se puede evitar que el usuario se desoriente
- El sistema web tendrá botones de navegación estándar (ingresar, salir, cerrar sesión, etc.)
- El sistema web no permitirá que se abran ventanas emergentes, con el fin de que el usuario no se desoriente.

#### **c) Requisitos de Usabilidad**

- El sistema web se visualizará en una interfaz que se vea bien y sea fácil de usar.
- El sistema web presentará mensajes de error en todos los casos que se necesite.
- El sistema web presentará solo los datos que son útiles para el usuario, de manera óptima.
- El sistema web no autorizará al usuario ejecutar procesos que no han sido concluidos.

#### **d) Requisitos de Escalabilidad**

- El sistema web podrá ser actualizado fácilmente sin alterar las funcionalidades ni módulos que contiene.

#### **e) Requisitos Operacionales**

- El sistema web tendrá campos obligatorios.
- El sistema web tendrá validaciones de contraseñas para que los usuarios puedan acceder al sistema.

**f) Requisitos de Seguridad**

- El sistema web se asegurará que la información este accesible únicamente al usuario autorizado.
- El sistema web no difundirá la información personal de ningún usuario.

**g) Requisitos Hardware**

- Para el uso del sistema web no existe ninguna restricción en lo que se refiere a hardware, porque no se debe realizar ninguna instalación en el dispositivo en el cual el usuario vaya a usar el sistema.
- El sistema web debe permitir ser visualizada en las diferentes resoluciones de pantalla.

## CAPÍTULO 4

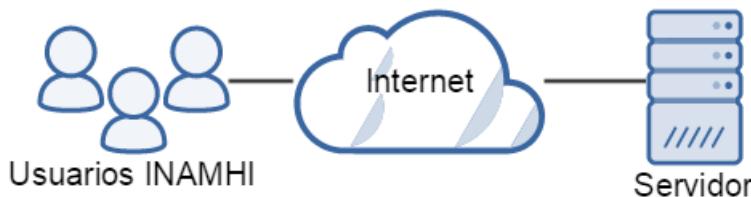
### FASE CODIFICACIÓN, IMPLEMENTACIÓN Y PRUEBAS

#### **4.1 Desarrollo del prototipo**

Para la construcción del sistema web para gestión de inventario de elementos hidrometeorológicos para el Instituto Nacional de Meteorología e Hidrología se hizo uso de las tecnologías Java EE.

Se desarrolló una sola aplicación integrada, en la que solo se permitirá el acceso a los usuarios registrados en el sistema y a las tareas a las cuales tengan acceso autorizado. Se empleó un solo servidor centralizado.

La Figura 13 muestra una representación de la arquitectura utilizada para el desarrollo de la aplicación web.



**Figura 13. Representación de la arquitectura.**

##### **4.1.1 Tecnologías JAVA EE**

En el presente proyecto se hace uso de Java EE, Java Enterprise Edition, siendo una plataforma que permite tanto el desarrollo como la ejecución de software. Entre las tecnologías que ofrece esta plataforma, se hizo uso de las siguientes:

**i. JSF (Java Server Faces):**

La aplicación web se realizó con JSF 2.2 un marco de trabajo de interfaces de usuario basado en una Arquitectura de Modelo Vista que separa sus componentes para ofrecer mayor control sobre cada parte de la aplicación facilitando su desarrollo y mantenimiento.

**ii. PrimeFaces:**

Para la creación y diseño de la aplicación web se hizo uso del framework PrimeFaces en su versión más reciente 5.2 que contiene componentes visuales de código abierto para el conjunto Java Server Faces 2.2.

**iii. Java Persistence API (JPA):**

Proporciona un modelo de persistencia para mapear bases de datos relacionales. JPA es utilizado para acceder y manipular datos relacionales de Enterprise Java Beans (EJBs).

#### **iv. Maven**

Gestiona las librerías y sus versiones a través del concepto de Artefacto. Un Artefacto puede verse como una librería pero agrupa más conceptos. Contiene las clases propias de la librería y además incluye toda la información necesaria para su correcta gestión (grupo, versión, dependencias, etc.).

#### **v. Spring**

Es un grupo de clases que manejan la infraestructura de desarrollo de una aplicación java, funde todos los componentes del proyecto, gestiona su ciclo de vida y como interactúan entre ellos.

#### **vi. Hibernate**

Automatiza la creación de objetos en un proyecto que representan tablas de una base de datos relacional, posibilita la configuración en archivos XML o anotaciones en los beans de las entidades que facultan establecer estas relaciones.

#### **vii. Java Naming and Directory Interface (JNDI)**

Proporcionan accesos a nombres y directorios.

#### **viii. Java Servlets**

Son clases de java utilizadas comúnmente para ampliar las capacidades de un servidor, también examina las peticiones, refresca la interfaz cliente y los datos de los beans.

### **4.1.2 Arquitectura del desarrollo**

Uno de los procesos más importantes dentro de la construcción del software es el desarrollo de la arquitectura de software, que permite representar la estructura del sistema con sus diferentes aspectos, además de ayudar a realizar el análisis para la toma de decisiones.

En el desarrollo del presente proyecto se utilizó la arquitectura por capas, siendo una arquitectura robusta que constituye un patrón fundamental para el desarrollo de software.

En la Tabla 5 se describe la estructura total del modelo de implementación, la descomposición del software en capas y los subsistemas de las mismas.

**Tabla 5.****Descomposición de capas del sistema**

CAPA	Objetos persistentes	Acceso Base de datos	Negocio	Vista		
				MVC		
DESCRIPCIÓN	Clases de mapeo de tablas de la base de datos	DAO	Servicios	Modelo	Vista	Controlador
				Beans	Páginas JSF Ficheros xhtml	Faces Servlets

**i. Objetos Persistentes**

Constituyen las clases de mapeo de tablas de la base de datos.

**ii. Base de Datos**

Está compuesta por las clases data object Access, los cuales son componentes de software que suministran una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos.

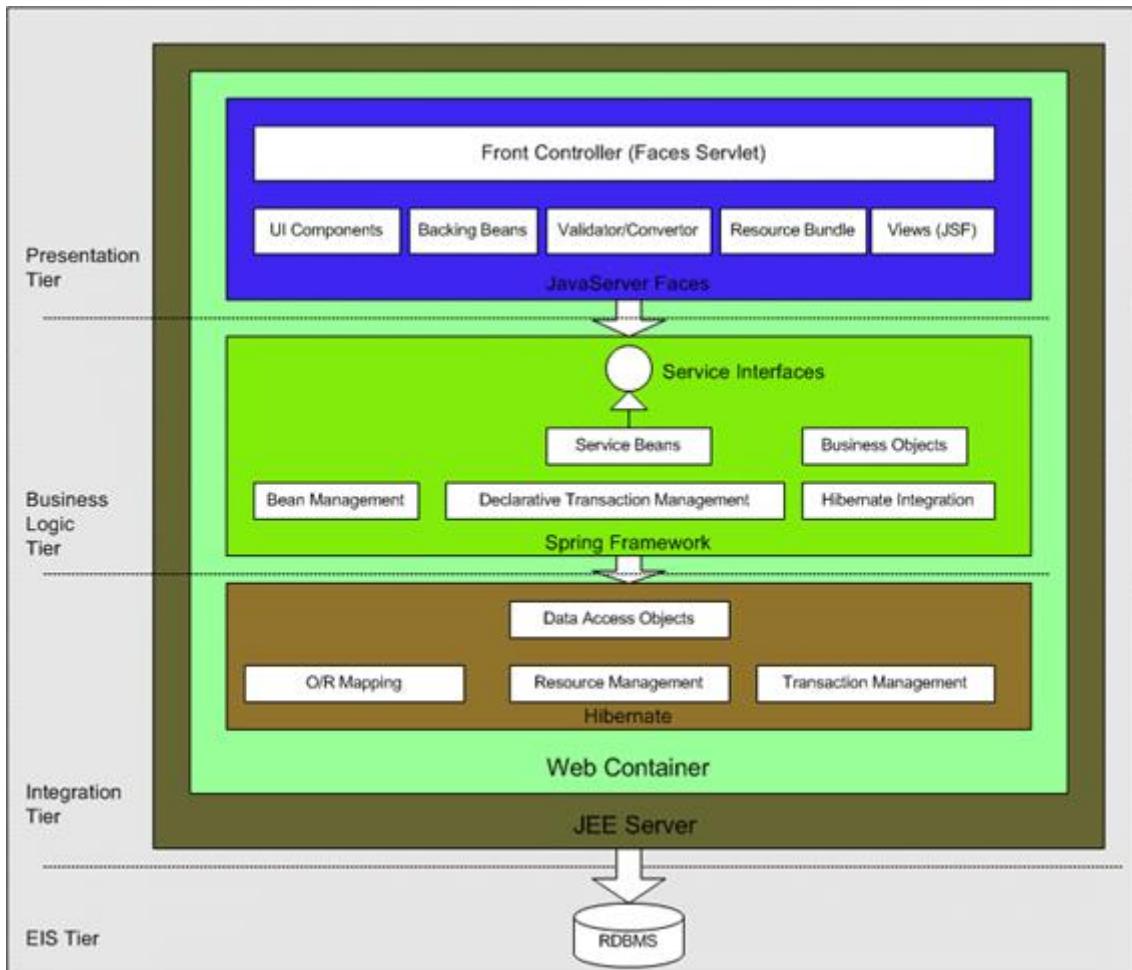
**iii. Capa Negocio**

Contiene los servicios que va a implementar el sistema como son: gestionar órdenes de trabajo, elementos y solicitudes de elementos.

**iv. Capa Vista**

La vista de implementación seguirá el patrón de diseño MVC (Modelo Vista Controlador). Este patrón plantea la separación del problema en tres capas: la capa modelo, la capa controlador y la capa vista.

La siguiente figura muestra el diseño de la arquitectura de alto nivel y cómo se acopla en la arquitectura multi-capas.



**Figura 25. Diseño de la arquitectura del sistema**

Fuente:(Suarez, 2011)

El Modelo Vista Controlador (MVC) separa la lógica del negocio con respecto a los datos (modelo) y la interfaz de usuario (vista/GUI). Permite modificaciones independientes en cada una las partes sin afectar a la otra. En otras palabras, los cambios realizados en la interfaz de usuario (GUI) no afectan el manejo de datos, y los datos pueden ser reorganizados sin cambiar la interfaz de usuario.

A continuación se describen los componentes de esta arquitectura:

### a) Modelo

Este componente se define mediante los beans gestionados, un bean es una clase con un conjunto de atributos o propiedades, los métodos getters y setters para devolver y actualizar valores y además varios otros métodos destinados a manejar comportamientos que queremos ver reflejados en la vista.

En la Figura 19 se muestra un fragmento de código de una clase modelo:

```

package com.inamhi.beans;
import com.inamhi.beans.util.JsfUtil;
import com.inamhi.entidades.Clases;
import com.inamhi.servicios.ClasesService;
import java.io.Serializable;
import java.util.Date;
import java.util.List;
import java.util.ResourceBundle;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.ejb.EJBException;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Controller;
@Controller
@Scope("session")
public class ClasesBean extends GenericBean<Clases> implements Serializable {
    @Autowired
    private ClasesService clasesService;
    @Autowired
    private LoginBean loginBean;
    public ClasesBean() {
        super(Clases.class);
        System.out.println("Estoy en el Bean Clases");
    }
    @Override
    public void persist(PersistAction persistAction, String successMessage) {
        if (this.getSelected() != null) {
            try {
                if (persistAction != ClasesBean.PersistAction.DELETE) {
                    if (persistAction == ClasesBean.PersistAction.CREATE) {
                        getSelected().setClasesusing(loginBean.getUsername());
                        this.getSelected().setClasesfing(new Date());
                        this.getSelected().setClaseselimin(Boolean.FALSE);
                        System.out.println("Estoy en el Bean clasesBean en el if CREATE");
                    } else {
                        getSelected().setClasesumod(loginBean.getUsername());
                        this.getSelected().setClasesfmod(new Date());
                        this.getSelected().setClaseselimin(Boolean.FALSE);
                        System.out.println("Estoy en el Bean ClasesBean en el if UPDATE");
                    }
                }
            } catch (Exception e) {
                FacesUtil.addErrorMessage(e.getMessage());
            }
        }
    }
}

```

**Figura 26. Fragmento de código de una clase bean**

### b) Controlador

Es implementado por los java servlets por medio de la clase facesservlet la cual maneja las acciones del usuario realizadas en la Vista y presenta la información correspondiente en la misma.

En la Figura 20 se muestra la configuracion de faces servlet en el descriptor de despliegue o archivo web.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>
            /WEB-INF/applicationContext.xml
            /WEB-INF/applicationContext-security.xml
        </param-value>
    </context-param>

    <servlet>
        <servlet-name>Faces Servlet</servlet-name>
        <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>*.xhtml</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <servlet>
        <servlet-name>Resource Servlet</servlet-name>
        <servlet-class>org.primefaces.resource.ResourceServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Resource Servlet</servlet-name>
        <url-pattern>/primefaces_resource/*</url-pattern>
    </servlet-mapping>
    <filter>
        <filter-name>springSecurityFilterChain</filter-name>
        <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
    </filter>

    <filter-mapping>
        <filter-name>springSecurityFilterChain</filter-name>
        <url-pattern>/*</url-pattern>
        <dispatcher>REQUEST</dispatcher>
    </filter-mapping>
</web-app>

```

**Figura 27. Configuración de faces servlet**

### c) Vista

Conforma la jerarquía de archivos xhtml que describen cada una de las páginas (pantallas) del interfaz de usuario de la aplicación. Se encarga de responder todas las peticiones de los beans vinculándolas con los componentes JSF.

En la Figura 21 se muestra un fragmento de código de una vista:



```

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:p="http://primefaces.org/ui">
    <p:dialog id="MarcEditDlg" widgetVar="MarcEditDialog" modal="true" resizable="false" appendTo="@body">
        <script>
            function handleSubmitEdit(xhr, status, args) {
                if (args.validationFailed)
                {
                    PF('MarcEditDialog').show();
                }
                else {
                    PF('MarcEditDialog').hide();
                }
            }
        </script>
        <h:form class="formColumn" id="MarcEditForm">
            <h:panelGroup id="display">
                <p:panelGrid columns="2" rendered="#{marcBean.selected != null}">
                    <h:outputLabel value="#{bundle.EditMarcLabel_nombre}" for="nombre" />
                    <h:inputText id="nombre" value="#{marcBean.selected.marcnomb}" required="true" />

                    <h:outputLabel value="#{bundle.EditMarcLabel_descripcion}" for="descripcion" />
                    <h:inputText id="descripcion" value="#{marcBean.selected.marcdesc}" required="true" />

                </p:panelGrid>
                <p:commandButton styleClass="botonAzul" actionListener="#{marcBean.save}" value="#{bundle.Save}"
                    update="display,:MarcListForm:marc,:growl" oncomplete="handleSubmitEdit(xhr,status,args)" />
                <p:commandButton value="#{bundle.Cancel}" onclick="PF('MarcEditDialog').hide();"/>
            </h:panelGroup>
        </h:form>
    </p:dialog>
</ui:composition>

```

**Figura 28. Fragmento de código de un archivo xhtml**

#### 4.1.3 Paquetes de diseño

##### i. DAO

Se compone de las clases que el proyecto usa para gestionar la persistencia de los objetos con la Base de Datos.

##### ii. Servicio

Contiene la clase servicio en la que se encuentran los servicios tales como insertar, eliminar, buscar. También tiene la clase implementación del servicio en la cual se implementan los servicios.

##### iii. Beans

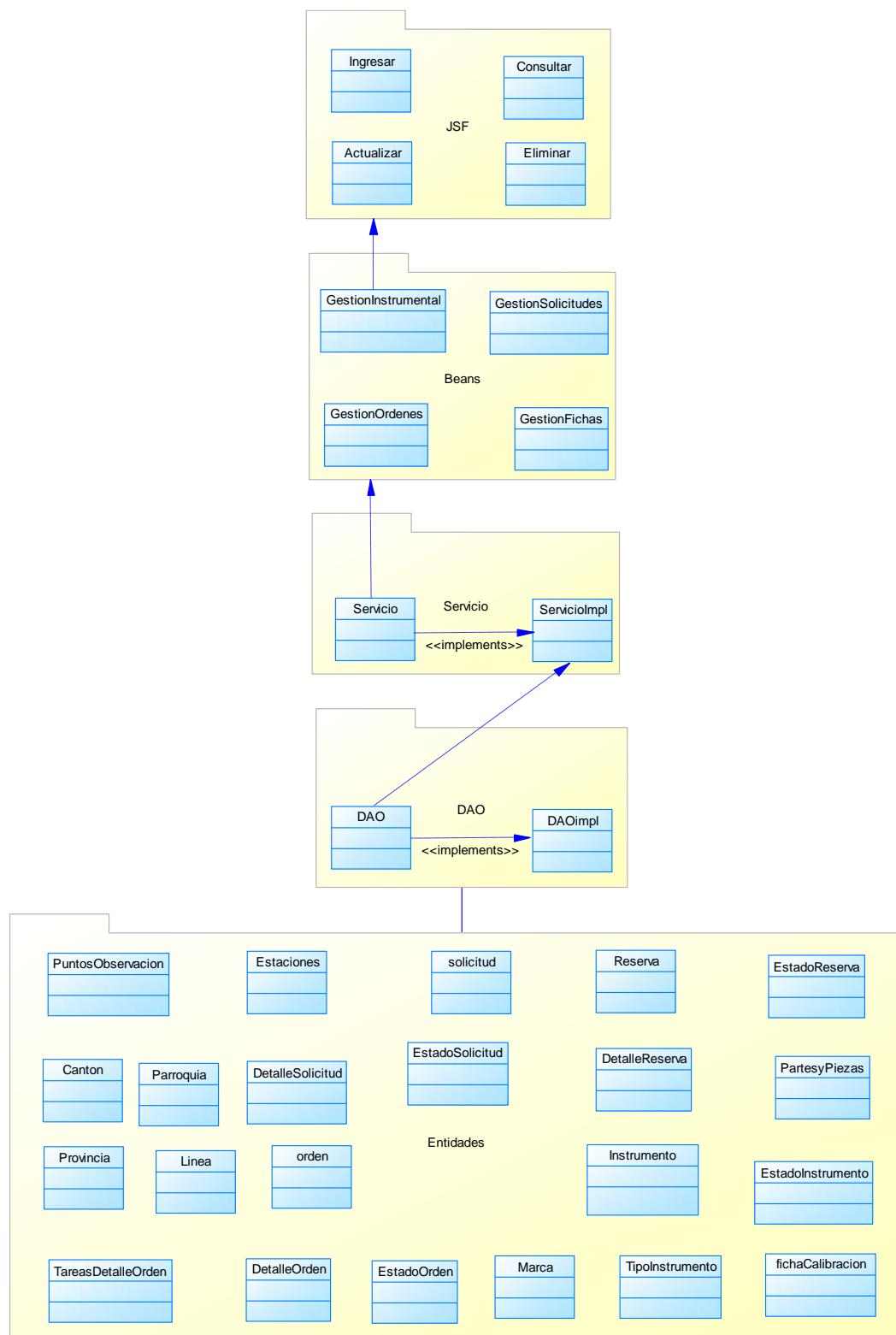
Contiene las clases donde están definidas las reglas del negocio.

#### iv. JSF

Contiene las páginas que forman la vista del sistema.

#### v. Entidades

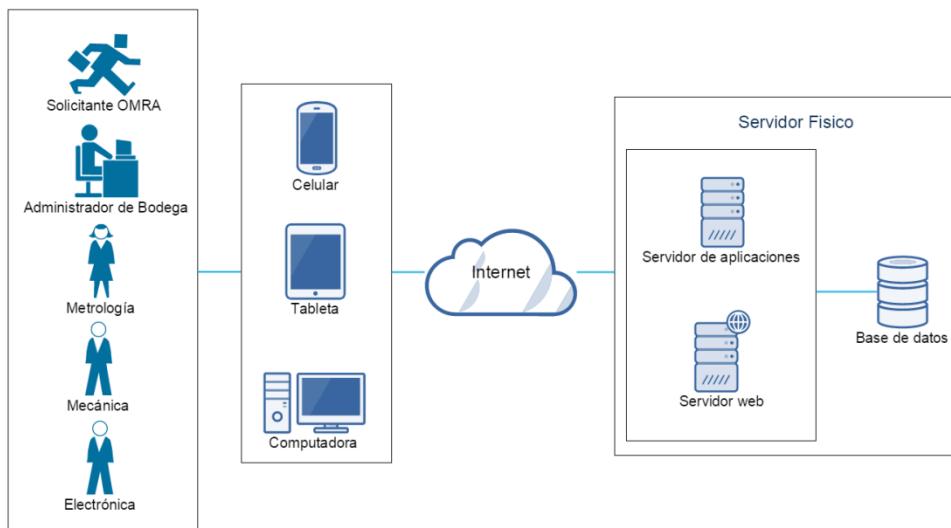
Cada entidad es una clase que representa una tabla de la base de datos.



**Figura 29. Paquetes de diseño**

#### 4.1.4 Vista de despliegue

El presente Diagrama de Despliegue muestra de manera gráfica los nodos que conforman el sistema. Cada nodo representa un recurso de ejecución como una computadora o servidor y se encuentra conectado con otros mediante un enlace de comunicación.



**Figura 30. Diagrama físico de las redes que utiliza el sistema**

##### i. Servidor de aplicaciones

Se utilizará el servidor glassfish el cual implementa las tecnologías definidas en la plataforma Java EE y permite ejecutar aplicaciones que siguen esta especificación.

##### ii. Base de datos

Se utilizara la base de datos postgresql, en la cual se almacenarán los datos del sistema.

#### 4.1.5 Módulos del sistema

Los módulos elaborados para cumplir con los requerimientos son los siguientes:

##### a) Gestión de usuarios

Gestiona cuentas de usuario en las cuales cada usuario puede ver, crear, editar o eliminar solo la información que le está asignada al rol al que pertenece.



**Figura 31. Login de usuario**

Este módulo muestra una lista con todos los usuarios

**Figura 32. Lista de usuarios**

Al dar click en el botón Crear se puede ingresar un nuevo usuario con sus respectivos roles

Nombre	<input type="text"/>
Correo	<input type="text"/>
Teléfono	<input type="text"/>
Área	<input type="text"/>
Observación	<input type="text"/>
Nombre de Usuario	<input type="text"/>
Contraseña	<input type="password"/>
Reescribir Contraseña	<input type="password"/>
Habilitado	<input checked="" type="checkbox"/>

Guardar Cancelar

**Roles**  
 10 ▾  
 Roles  
 No records found.  
 + Añadir Borrar Todo

**Figura 33. Formulario para usuario nuevo**

Al dar clic en Editar se muestra un formulario con las propiedades que tiene el usuario, otorgando la opción de borrar y sobrescribir la propiedad que se deseé y se guardará en la base de datos cuando se de clic en la opción guardar.

Nombre	Jacqueline De La Cruz
Correo	jdelacruz@inamhi.gob.ec
Teléfono	3971100
Área	Desarrollo
Observación	<input type="text"/>
Nombre de Usuario	jdelacruz
Contraseña	*****
Reescribir Contraseña	*****
Habilitado	<input checked="" type="checkbox"/>

Guardar Cancelar

**Roles**  
 10 ▾  
 Roles  
 No records found.  
 + Añadir Borrar Todo

**Figura 34. Formulario para edición de usuario**

## b) Gestión de Elementos

Este módulo muestra dos opciones en la primera denominada ingreso se mostrará una lista con todos los elementos con sus más relevantes características.

Código	Línea	Nombre	Marca	Modelo	Número de Serie	Código Inamhi	Código QR
11	DATALOGER	4	LOGOTRONIC	4	4	593	
12	DATALOGER	Janeth Roldan	CSB	rgerterwerterwetert	12	593	
21	SENSOR	1	VAISALA	1	1	1	

**Figura 35. Lista de elementos**

Al dar clic en el botón crear se abrirá un formulario vacío en el cual se podrán ingresar propiedades para crear un elemento nuevo

**Figura 36. Formulario para elemento nuevo**

Seleccionando cualquier elemento y dando clic en editar se abrirá un formulario con las propiedades actuales del elemento, las cuales se podrán borrar y sobreescibir y luego se guardarán en la base de datos cuando se presione el botón guardar.

**Figura 37. Formulario para edición de elemento**

En el botón editar se podrán actualizar características propias de cada elemento.

**Figura 38. Formulario para edición de propiedades de elemento**

Al abrir la pestaña Mantenimiento se desplegaran las siguientes opciones:

### i. Bodegas

Muestra la lista de bodegas

The screenshot shows the INAMHI inventory management system interface. The left sidebar has a green header 'Gestionar Elementos' and a 'Mantenimiento' section with 'Bodegas' selected. The main area is titled 'Gestión de Bodegas' and displays a table with two rows of data. The columns are: Bodega (1, 2), Nombre (Quito, Guayaquil), Bodega Ingresada (jescobar, jescobar), Fecha Ingreso (2016-03-07 15:37:59.067, 2016-03-07 15:38:11.954), Usuario Modificacion (empty), and Fecha Modificacion (empty). Below the table are buttons for '+ Crear', 'Editar', and 'Eliminar'. The top right has a 'Cerrar Sesión' button.

Bodega	Nombre	Bodega Ingresada	Fecha Ingreso	Usuario Modificacion	Fecha Modificacion
1	Quito	jescobar	2016-03-07 15:37:59.067		
2	Guayaquil	jescobar	2016-03-07 15:38:11.954		

**Figura 39. Lista de Bodegas**

Al dar clic en el botón crear se abrirá un formulario para crear una nueva bodega.

The screenshot shows the same INAMHI inventory management system interface as Figure 39, but with a modal dialog box overlaid. The dialog is titled 'Crear Nueva Bodega' and contains a single input field labeled 'Nombre' with the value 'Quito'. At the bottom of the dialog are 'Guardar' and 'Cancelar' buttons. The rest of the interface remains the same, showing the list of warehouses and the sidebar menu.

**Figura 40. Formulario para bodega nueva**

Dando clic sobre cualquier bodega se la seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información de la bodega seleccionada, la cual se podrá modificar y luego se guardará dando clic en el botón guardar.

The screenshot shows the INAMHI inventory management interface. On the left, there is a sidebar with navigation links: 'Gestionar Usuarios', 'Ingresar', 'Gestionar Elementos' (selected), 'Ingresar', 'Mantenimiento' (expanded), 'Bodegas', 'Estado', 'Catálogo de Propiedades', 'Propiedades', 'Clases', 'Métodos de Medidas y Observación', 'Tipo', 'Marcas', 'Líneas', 'Gestionar Solicitud de Elementos', and 'Gestionar Órdenes de Trabajo'. The main content area is titled 'Gestión de Bodegas' and displays a table with columns: Bodega, Nombre, Bodega Ingresada, Fecha Ingreso, Usuario Modificación, and Fecha Modificación. Two rows are visible: row 1 has Bodega '1' and Nombre 'Quito'; row 2 has Bodega '2' and Nombre 'Guayaquil'. A modal dialog box titled 'Editar Bodega' is open over the table, containing a 'Nombre' input field with 'Quito' and two buttons: 'Guardar' (blue) and 'Cancelar' (white). At the bottom right of the modal is a red button labeled 'Eliminar'.

**Figura 41. Formulario para edición de bodega**

Al dar clic en el botón eliminar se borrará la bodega seleccionada y se mostrará un mensaje.

The screenshot shows the same INAMHI inventory management interface as Figure 41. The 'Gestionar Elementos' link in the sidebar is still selected. The main content area shows the 'Gestión de Bodegas' table with the same two rows as before. A yellow message box in the top right corner says 'Bodega Eliminada' with an information icon. At the bottom of the table, there are three buttons: '+ Crear', 'Editar' (with a pencil icon), and 'Eliminar' (in red).

**Figura 42. Eliminación de bodega**

## ii. Estado

Muestra la lista de estados de los elementos

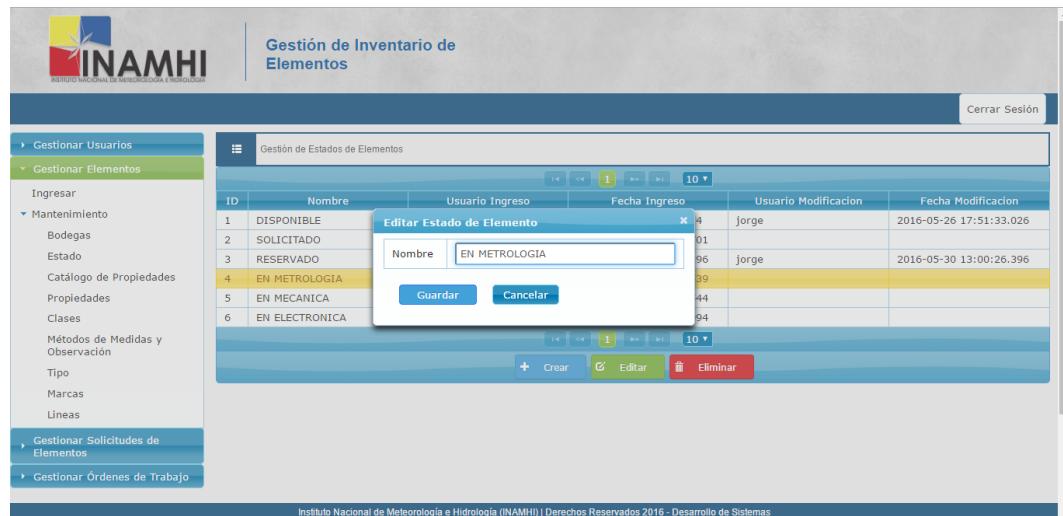
ID	Nombre	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	DISPONIBLE	jescobar	2016-03-07 15:44:28.14	jorge	2016-05-26 17:51:33.026
2	SOLICITADO	jorge	2016-05-30 13:00:26.396		
3	RESERVADO	jorge	2016-06-01 12:48:26.539	jorge	2016-05-30 13:00:26.396
4	EN METROLOGIA	jorge	2016-06-01 12:48:59.544		
5	EN MECANICA	jorge	2016-06-01 12:49:09.094		
6	EN ELECTRONICA	jorge	2016-06-01 12:49:09.094		

**Figura 43. Lista de estados de elementos**

Al dar clic en el botón crear se abrirá un formulario para crear un nuevo estado de elemento.

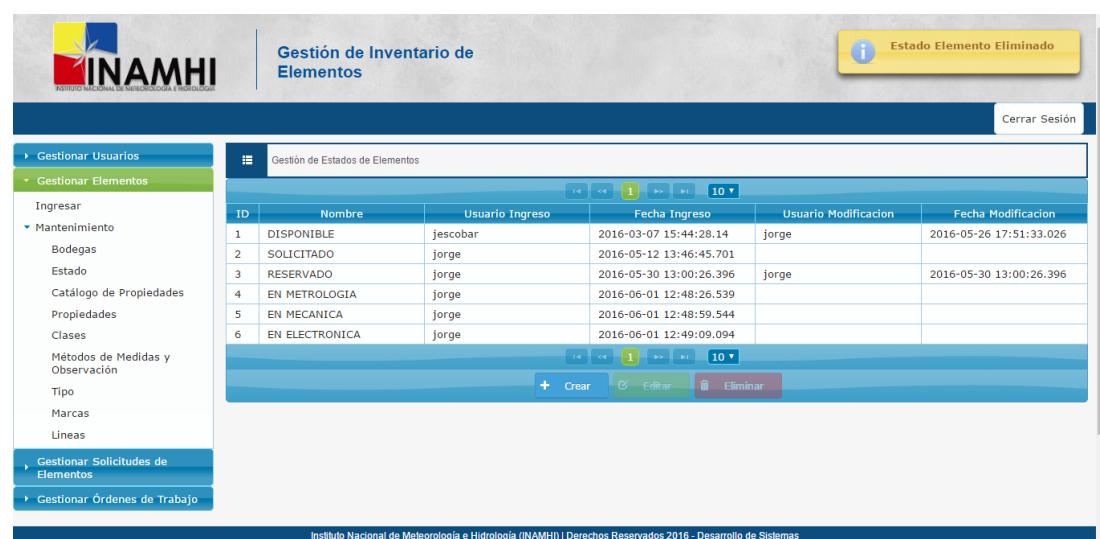
**Figura 44. Formulario para estado de elemento nuevo**

Dando clic sobre cualquier estado se lo seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información del estado de elemento seleccionado, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 45. Formulario para editar estado de elemento**

Al dar clic en el botón eliminar se borrará el estado de elemento seleccionado y se mostrara un mensaje.



**Figura 46. Eliminación de estado de elemento**

### iii. Catálogo de Propiedades

Muestra el catálogo de propiedades de elementos.

ID	Línea	Nombre	Descripción	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	TARJETA SIM	IP	IP de la tarjeta SIM	jorge	2016-03-11 14:43:29.326	jorge	2016-03-11 14:45:40.021
2	CAJA NEMA	Precableada	Si esta precableada	jorge	2016-03-11 14:46:10.352		
3	SENSOR	CABLE	Si incluye cable	jorge	2016-03-11 14:46:54.213		
4	TARJETA SIM	ICC		jorge	2016-03-11 14:50:38.842		
5	DATALOGER	Modulo Conectado		jorge	2016-03-17 13:01:05.953		
6	DATALOGER	Tipo de Tarjeta		jorge	2016-03-17 13:01:23.064		
7	BATERIA	Cargada	si esta cargada o no	jorge	2016-06-07 11:26:10.61	jorge	2016-06-07 11:26:18.181

Figura 47. Catálogo de propiedades de elementos

Al dar clic en el botón crear se abrirá un formulario para crear una nueva propiedad de elemento.

ID	Línea	Nombre	Descripción	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	TARJETA SIM			2016-03-11 14:43:29.326	jorge	2016-03-11 14:45:40.021	
2	CAJA NEMA			2016-03-11 14:46:10.352			
3	SENSOR			2016-03-11 14:46:54.213			
4	TARJETA SIM			2016-03-11 14:50:38.842			
5	DATALOGER	Modulo Conectado		2016-03-17 13:01:05.953			
6	DATALOGER	Tipo de Tarjeta		2016-03-17 13:01:23.064			
7	BATERIA	Cargada	si esta cargada o no	2016-06-07 11:26:10.61	jorge	2016-06-07 11:26:18.181	

Figura 48. Formulario para crear propiedad de elemento nueva

Para seleccionar una propiedad de elemento se tiene que dar clic sobre la misma. Al dar clic en el botón editar se abrirá un formulario con la información de la propiedad de elemento seleccionada, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 49. Formulario para editar propiedad de elemento**

Al dar clic en el botón eliminar se borrara la propiedad de elemento seleccionada y se mostrara un mensaje.



**Figura 50. Eliminación de propiedad de elemento**

#### iv. Propiedades

Muestra las propiedades individuales de los elementos.

ID	Nombre	Valor
1	IP	192.168.1.1
2	ICC	3453
3	IP	324
4	ICC	23
5	Modulo Conectado	no
6	Tipo de Tarjeta	sim

**Figura 51. Lista de Propiedades de elementos por propiedad**

Al dar clic en el botón crear se abrirá un formulario para crear un nuevo valor para una propiedad de elemento.

**Figura 52. Formulario para crear un nuevo valor para una propiedad de elemento**

Dando clic sobre cualquier propiedad de elemento se la seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información del valor de la propiedad de elemento seleccionada, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 53. Formulario para editar el valor de una propiedad de elemento**

Al dar clic en el botón eliminar se borrará el valor de la propiedad de elemento seleccionada y se mostrara un mensaje.



**Figura 54. Eliminación del valor de una propiedad de elemento**

## v. Clases

Muestra las clases de los elementos.

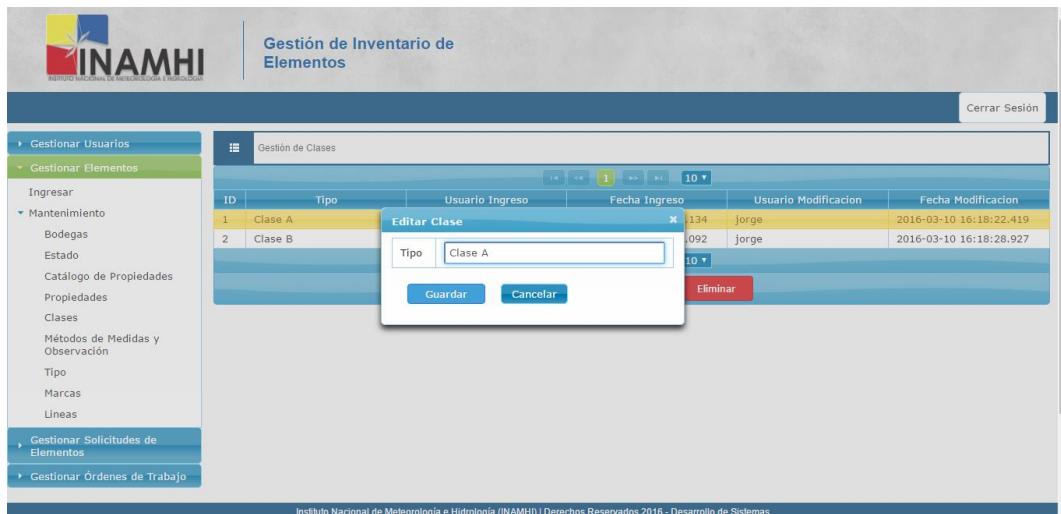
ID	Tipo	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	Clase A	jescobar	2016-03-07 15:39:13.134	jorge	2016-03-10 16:18:22.419
2	Clase B	jescobar	2016-03-07 15:42:35.092	jorge	2016-03-10 16:18:28.927

**Figura 55. Lista de clases de elementos**

Al dar clic en el botón crear se abrirá un formulario para crear una nueva clase de elemento.

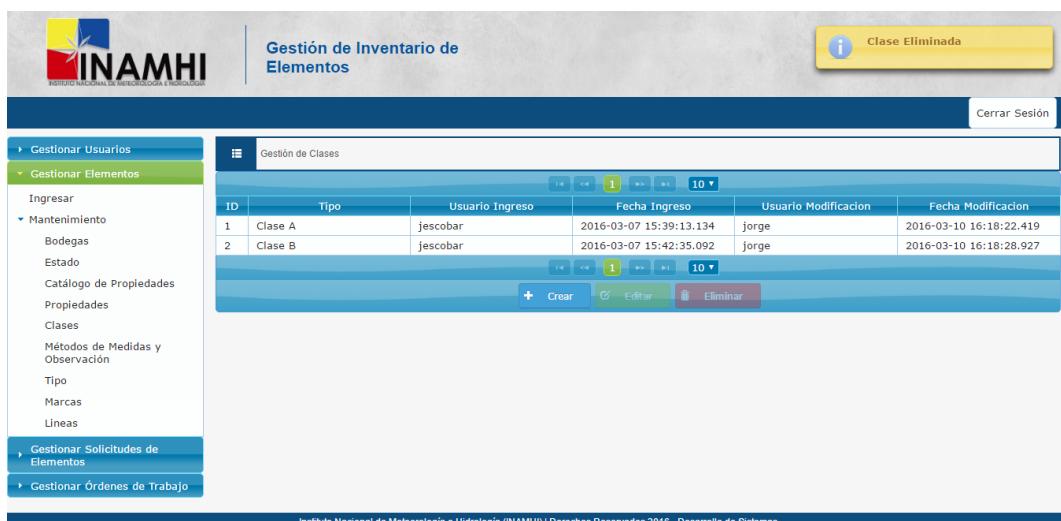
**Figura 56. Formulario para crear una clase nueva**

Dando clic sobre cualquier clase se la seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información del valor de la clase seleccionada, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 57. Formulario de edición de clase**

Al dar clic en el botón eliminar se borrara la clase seleccionada y se mostrara un mensaje.



**Figura 58. Eliminación de clase**

## vi. Métodos de Medidas y Observación

Muestra los métodos de medidas y observación de los elementos.

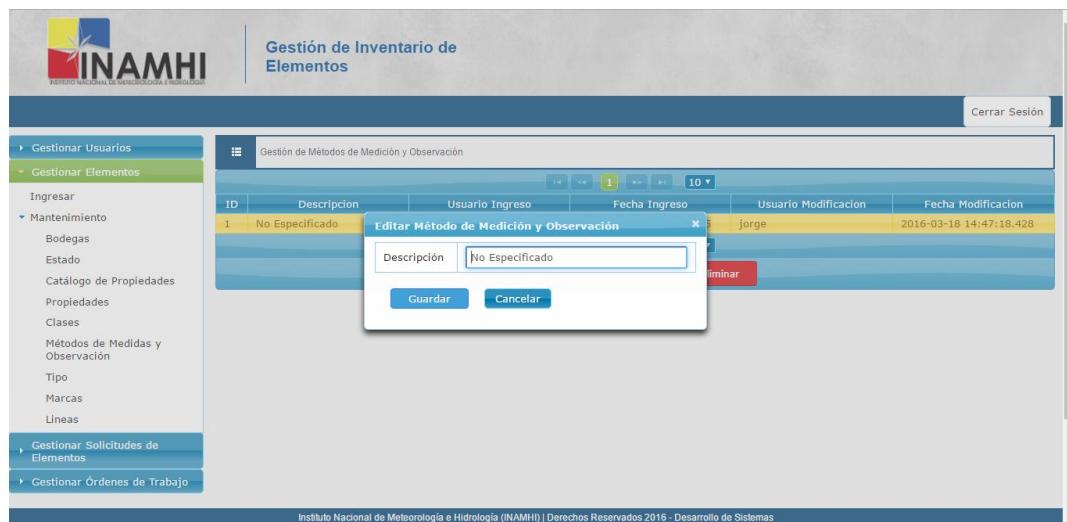
ID	Descripción	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	No Especificado	jescobar	2016-03-07 15:55:28.695	jorge	2016-03-18 14:47:18.428

**Figura 59. Lista de métodos de medición y observación**

Al dar clic en el botón crear se abrirá un formulario para crear un nuevo método de medida u observación de elemento.

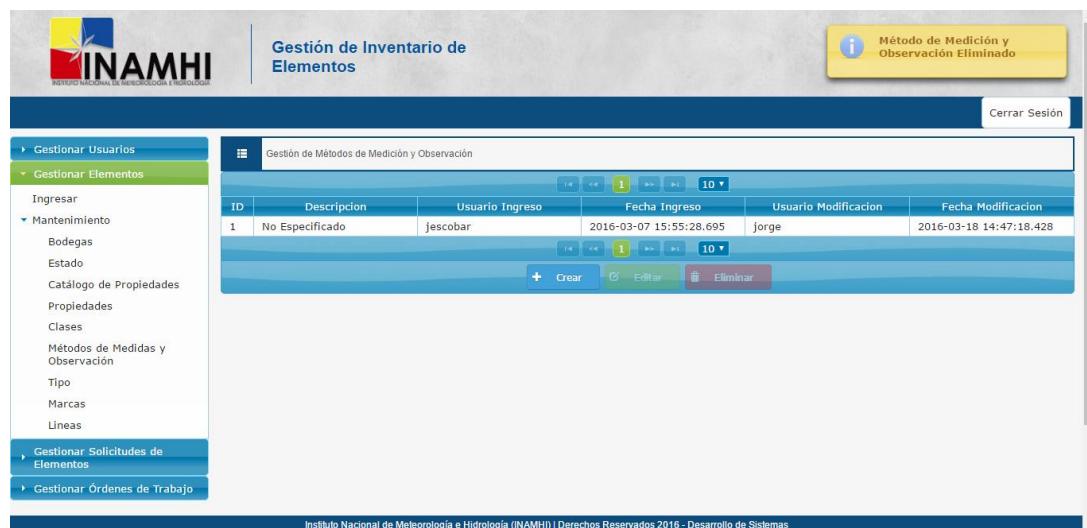
**Figura 60. Formulario para crear método de medición u observación**

Dando clic sobre cualquier método de medida u observación se lo seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información del valor del método de medida u observación seleccionado, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 61. Formulario de edición de método de medición u observación**

Al dar clic en el botón eliminar se borrara el método de medida u observación seleccionada y se mostrara un mensaje.



**Figura 62. Eliminación de método de medición u observación**

## vii. Tipo

Muestra los tipos de elementos.

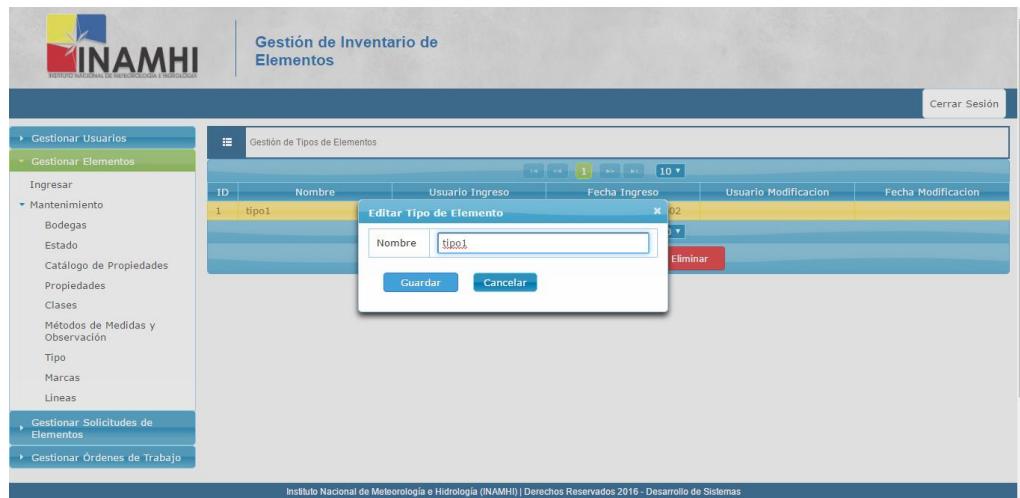
ID	Nombre	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	tipo1	jescobar	2016-03-07 15:56:16.602		

**Figura 63. Lista de tipos de elementos**

Al dar clic en el botón crear se abrirá un formulario para crear un nuevo tipo de elemento.

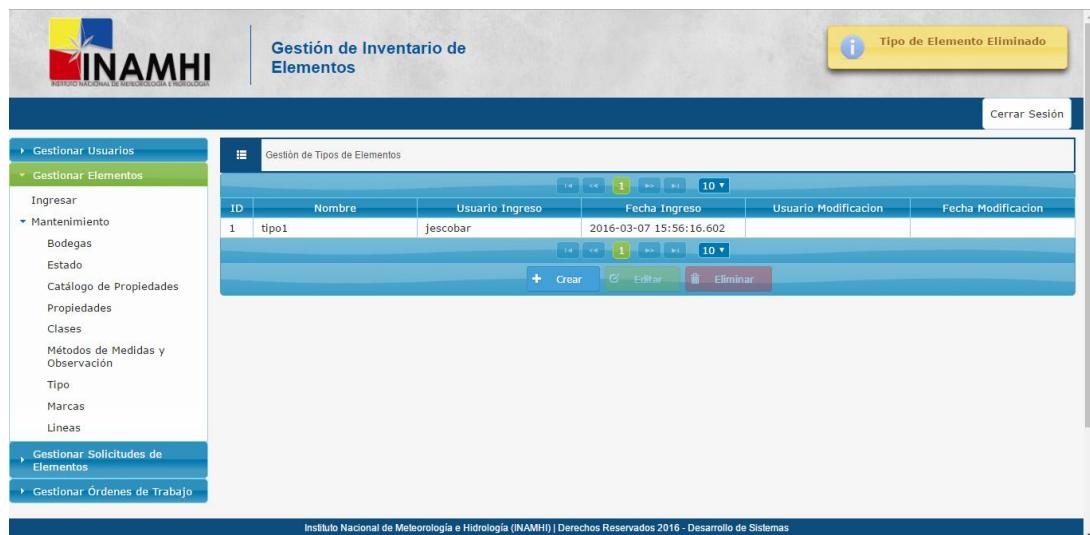
**Figura 64. Formulario para crear tipo de elemento nuevo**

Dando clic sobre cualquier tipo de elemento se lo seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información del tipo de elemento seleccionado, el cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 65. Formulario de edición de tipo de elemento**

Al dar clic en el botón eliminar se borrará el tipo de elemento seleccionado y se mostrara un mensaje.



**Figura 66. Eliminación de tipo de elemento**

### viii. Marcas

Muestra las marcas de elementos.

ID	Nombre	Descripción	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	NO ESPECIFICADO	NO ESPECIFICADO	ADMIN	2013-09-17 11:49:40.0		
2	VAISALA		ADMIN	2013-09-17 11:49:40.0		
3	CAMPBELL		ADMIN	2013-09-17 11:49:40.0		
4	WATCHING		ADMIN	2013-09-17 11:49:40.0		
5	LOGOTRONIC		ADMIN	2013-10-25 08:25:13.0		2013-10-25 08:25:13.0
6	KIPP & ZONNEN		ADMIN	2013-09-17 11:49:40.0		
7	CSB		ADMIN	2013-09-17 11:49:40.0		
8	MET ONE		ADMIN	2013-09-17 11:49:40.0		
9	NOVALINX		ADMIN	2013-09-17 11:49:40.0		
10	DUNCAN		ADMIN	2013-09-17 11:49:40.0		

Figura 67. Lista de marcas de elementos

Al dar clic en el botón crear se abrirá un formulario para registrar en el sistema una nueva marca de elemento.

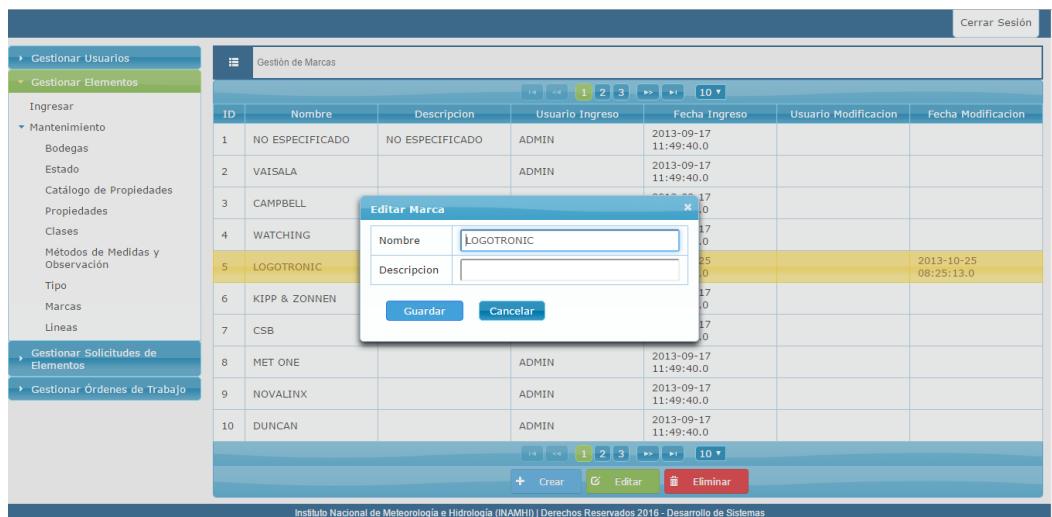
Nombre:

Descripción:

Guardar
Cancelar

Figura 68. Formulario para crea una marca nueva

Dando clic sobre cualquier marca de elemento se la seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información de la marca de elemento seleccionada, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 69. Formulario de edición de marca**

Al dar clic en el botón eliminar se borrara la marca del elemento seleccionado y se mostrara un mensaje.



**Figura 70. Eliminación de marca**

## ix. Líneas

Muestra las líneas de elementos.

ID	Nombre	Descripción	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	DATALOGER		ADMIN	2013-09-26 16:00:37.0	ADMIN	2013-09-26 16:00:37.0
2	SENSOR		ADMIN	2013-09-26 16:01:08.0	ADMIN	2013-09-26 16:01:08.0
3	PANEL SOLAR		ADMIN	2013-09-26 16:01:31.0	ADMIN	2013-09-26 16:01:31.0
4	PLUVIOMETRO		ADM	2013-09-27 10:56:57.0	ADM	2013-09-27 10:56:57.0
5	CAJA NEMA		ADM	2013-09-27 11:40:07.0	ADM	2013-09-27 11:40:07.0
6	MODEM - ANTENA GPRS		ADM	2013-09-27 10:58:15.0	ADM	2013-09-27 10:58:15.0
7	PANTALLA SENSOR TEMP- HF		ADM	2013-09-27 11:02:23.0	ADM	2013-09-27 11:02:23.0
8	TARJETA SIM		ADM	2013-09-27 10:58:43.0	ADM	2013-09-27 10:58:43.0
9	FILTRO DE COMUNICACIONES		ADM	2013-09-27 10:59:03.0	ADM	2013-09-27 10:59:03.0
10	RELE		ADM	2013-09-27 10:59:17.0	ADM	2013-09-27 10:59:17.0

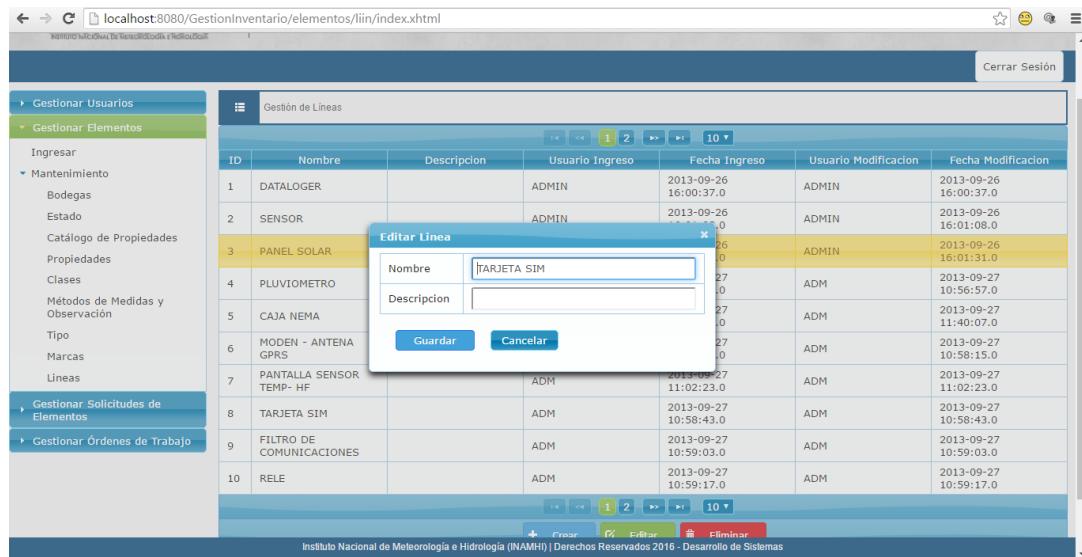
Figura 71. Lista de líneas

Al dar clic en el botón crear se abrirá un formulario para registrar en el sistema una nueva línea de elemento.

ID	Nombre	Descripción	Usuario Ingreso	Fecha Ingreso	Usuario Modificación	Fecha Modificación
1	DATALOGER		ADMIN	2013-09-26 16:00:37.0	ADMIN	2013-09-26 16:00:37.0
2	SENSOR		ADMIN	2013-09-26 16:01:08.0	ADMIN	2013-09-26 16:01:08.0
3	PANEL SOLAR		ADM	2013-09-26 16:01:31.0	ADM	2013-09-26 16:01:31.0
4	PLUVIOMETRO		ADM	2013-09-27 10:56:57.0	ADM	2013-09-27 10:56:57.0
5	CAJA NEMA		ADM	2013-09-27 11:40:07.0	ADM	2013-09-27 11:40:07.0
6	MODEM - ANTENA GPRS		ADM	2013-09-27 10:58:15.0	ADM	2013-09-27 10:58:15.0
7	PANTALLA SENSOR TEMP- HF		ADM	2013-09-27 11:02:23.0	ADM	2013-09-27 11:02:23.0
8	TARJETA SIM		ADM	2013-09-27 10:58:43.0	ADM	2013-09-27 10:58:43.0
9	FILTRO DE COMUNICACIONES		ADM	2013-09-27 10:59:03.0	ADM	2013-09-27 10:59:03.0
10	RELE		ADM	2013-09-27 10:59:17.0	ADM	2013-09-27 10:59:17.0

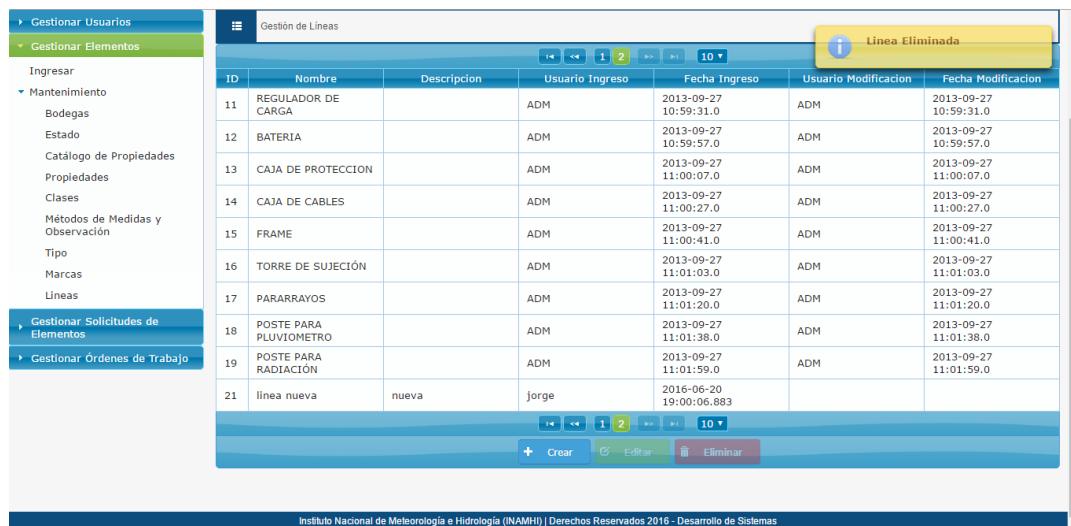
Figura 72. Formulario de creación de línea nueva

Dando clic sobre cualquier línea de elemento se la seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información de la línea de elemento seleccionada, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 73. Formulario de edición de línea**

Al dar clic en el botón eliminar se borrara la línea del elemento seleccionado y se mostrara un mensaje.



**Figura 74. Eliminación de línea**

### c) Gestión de las solicitudes de elementos

Administra la creación de solicitudes y el manejo del estado de las mismas así como de los elementos pedidos.

Al dar clic en ingresar se abrirá un formulario donde podremos crear una solicitud.

**Figura 75. Formulario para crear una nueva solicitud de elementos**

Al dar clic en el botón escoger se desplegará una lista en la cual se podrá escoger la estación para la cual se va a hacer la solicitud. Esto se puede hacer dando clic sobre la estación a seleccionar y luego dando clic en aceptar.

ID	Código	Nombre	Tipo
38	M1107	LAGUACOTO	CLIMATOLOGICA PRINCIPAL
63779	H1150	RÍO PATUL	VAISALA
63781	M1124	SIERRAZUL	VAISALA
63782	M0221	SAN CRISTOBAL-GALAPAGOS	VAISALA
63786	HS012	ATAPO PUMACHACA	SUTRON
63787	H5013	RÍO GUASUNTOS EN LA MOYA	SUTRON
63777	M0024	ÍÑAQUITO	VAISALA
63788	H5014	RÍO CHAMBO EN PENIPE	SUTRON
63789	H5015	RÍO CEBADAS DJ YASIPAN	SUTRON
63790	M1230	ESPM-MFL- CALCETA	LOGOTRONIC

**Figura 76. Lista de estación desplegable**

Cuando se da clic en el botón añadir elemento se desplegará una lista de los elementos disponibles los cuales se podrán pedir en la solicitud uno por uno dando clic sobre los mismos y luego presionando el botón aceptar.

**Figura 77. Lista de elementos disponibles para agregar a la solicitud**

Después de haber llenado todos los campos se podrá finalizar la solicitud dando clic en guardar solicitud o salir a la página principal dando clic en cancelar.

**Figura 78. Solicitud de elementos llena**

En la opción Asignar Estado podremos ver una lista de todas las solicitudes.

ID	Estación	Nombre Solicitante	Fecha de Solicitud	Fecha de Comisión	Estado de Solicitud
38_2016-05-12_00:00:00	LAGUACOTO	Jorge	2016-05-12 00:00:00.0	2016-05-31 00:00:00.0	NEGADA
63779_2016-05-12_00:00:00	RIO PATUL	Javier	2016-05-12 00:00:00.0	2016-05-22 00:00:00.0	APROBADA
63781_2016-05-12_00:00:00	SIERRAZUL	jake	2016-05-03 00:00:00.0	2016-05-20 00:00:00.0	APROBADA
63781_2016-05-12_00:00:00	SIERRAZUL	Jacqueline de la Cruz	2016-05-12 00:00:00.0	2016-06-30 00:00:00.0	NEGADA
63790_2016-06-16_00:00:00	ESPAM-MFL- CALCETA	jake	2016-06-16 00:00:00.0	2016-06-17 00:00:00.0	SOLICITADA

**Figura 79. Lista de solicitudes**

Dando clic sobre cualquier solicitud la seleccionaremos

ID	Estación	Nombre Solicitante	Fecha de Solicitud	Fecha de Comisión	Estado de Solicitud
38_2016-05-12_00:00:00	LAGUACOTO	Jorge	2016-05-12 00:00:00.0	2016-05-31 00:00:00.0	NEGADA
63779_2016-05-12_00:00:00	RIO PATUL	Javier	2016-05-12 00:00:00.0	2016-05-22 00:00:00.0	APROBADA
63781_2016-05-12_00:00:00	SIERRAZUL	jake	2016-05-03 00:00:00.0	2016-05-20 00:00:00.0	APROBADA
63781_2016-05-12_00:00:00	SIERRAZUL	Jacqueline de la Cruz	2016-05-12 00:00:00.0	2016-06-30 00:00:00.0	NEGADA
63790_2016-06-16_00:00:00	ESPAM-MFL- CALCETA	jake	2016-06-16 00:00:00.0	2016-06-17 00:00:00.0	SOLICITADA

**Figura 80. Solicitud seleccionada**

Al dar clic sobre el botón Mostrar Detalle podremos ver los elementos que se han pedido en la solicitud seleccionada.

The screenshot shows a web-based application interface for managing inventory requests. On the left, there is a sidebar with navigation links: 'Gestionar Elementos', 'Gestionar Solicitud de Elementos' (selected), 'Ingresar', 'Asignar Estado', 'Mantenimiento' (selected), 'Estado De Solicitud', and 'Gestionar Órdenes de Trabajo'. The main content area has two tables. The top table lists requests with columns: ID, Estación, Nombre Solicitante, Fecha de Solicitud, Fecha de Comisión, and Estado de Solicitud (with a dropdown menu). The bottom table lists items with columns: ID, Marca, Línea, Elemento, and Estado. Buttons at the bottom include 'Mostrar Detalle' (highlighted in green), 'Aprobar Todo' (grey), and 'Negar Todo' (red). The footer of the page reads 'Instituto Nacional de Meteorología e Hidrología (INAMHI) | Derechos Reservados 2016 - Desarrollo de Sistemas'.

**Figura 81. Elementos pedidos en una solicitud**

Se puede aprobar solo los elementos que se deseé dando clic sobre los mismos y dando clic en el botón aprobar.

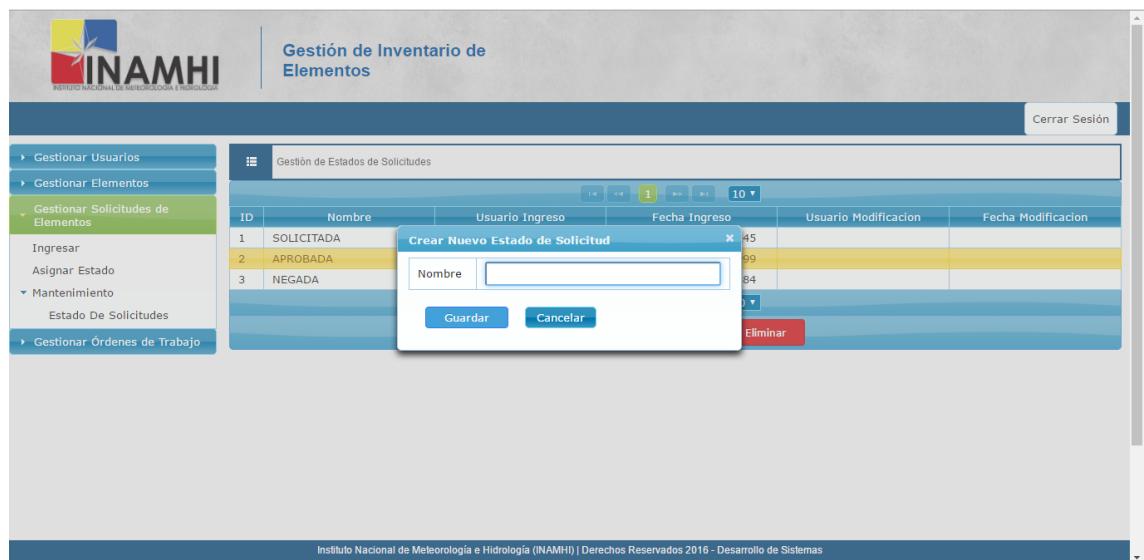
Para aprobar todos los elementos de la solicitud seleccionada se tiene que dar clic en el botón Aprobar Todo. O a su vez el botón “Negar Todo” negará todos los elementos de la solicitud seleccionada.

En la pestaña Mantenimiento encontramos la opción Estado de solicitudes en la cual se muestra los estados de solicitudes existentes.

The screenshot shows the 'Gestión de Estados de Solicitud' (Management of Request States) section. The sidebar includes 'Gestionar Usuarios', 'Gestionar Elementos' (selected), 'Gestionar Solicitud de Elementos', 'Ingresar', 'Asignar Estado', 'Mantenimiento' (selected), 'Estado De Solicitud' (selected), and 'Gestionar Órdenes de Trabajo'. The main content area displays a table titled 'Gestión de Estados de Solicitud' with columns: ID, Nombre, Usuario Ingreso, Fecha Ingreso, Usuario Modificación, and Fecha Modificación. The table contains three rows: 'SOLICITADA' (ID 1), 'APROBADA' (ID 2), and 'NEGADA' (ID 3). At the bottom are buttons for '+ Crear', 'Editar', and 'Eliminar'. The footer of the page reads 'Instituto Nacional de Meteorología e Hidrología (INAMHI) | Derechos Reservados 2016 - Desarrollo de Sistemas'.

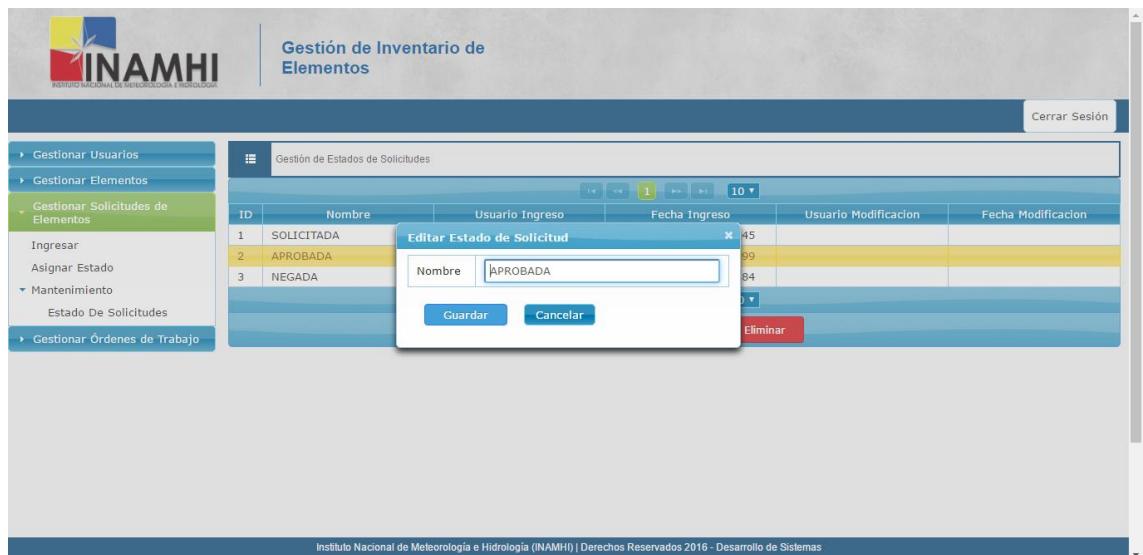
**Figura 82. Lista de estados de solicitudes**

Al dar clic en el botón crear se abrirá un formulario para registrar en el sistema un nuevo estado de solicitud.



**Figura 83. Formulario para crear estado de solicitud nuevo**

Dando clic sobre cualquier estado de solicitud se lo seleccionará. Al dar clic en el botón editar se abrirá un formulario con la información del estado de solicitud seleccionado, la cual se podrá modificar y luego se guardara dando clic en el botón guardar.



**Figura 84. Formulario de edición de estado de solicitud**

Al dar clic en el botón eliminar se borrará el estado de solicitud seleccionado y se mostrara un mensaje.

#### d) Gestión de órdenes de trabajo de elementos

Administra la creación de órdenes de trabajo y el manejo del estado de las mismas así como de los elementos vinculados a la orden.

Al dar clic en ingresar se abrirá un formulario donde podremos crear una orden de trabajo.

**Figura 85. Formulario para crear una orden de trabajo nueva**

El botón añadir elemento dirige a un formulario en el cual se podrá escoger el elemento y las tareas que se desea realizar en él. Se puede añadir los elementos que se desee.

**Figura 86. Formulario para escoger elementos y tareas que se requieren realizar**

Después de haber llenado todos los campos se finalizara la orden dando clic en el botón Guardar Orden.

The screenshot shows a web-based application for managing work orders. On the left, a sidebar menu under 'Gestionar Órdenes de Trabajo' includes 'Ingresar', 'Seguimiento de Órdenes', and 'Mantenimiento'. The main area is titled 'Ingreso de Órdenes' and contains fields for 'Área de la Orden' (selected as 'Escoja una...'), 'Fecha de solicitud' (21/06/16), 'Fecha de Entrega' (25/06/16), 'Solicitante' (Jorge Escobar), 'Área Solicitante' (DESARROLLO), and 'Observaciones'. Below this is a table with columns 'Línea', 'Marca', 'Modelo', 'Serie', 'Tareas', and 'Estado Elemento'. It lists two entries: 'SENSOR' by 'WATCHING' with model 12421, having tasks 'LIMPIEZA TOTAL' and 'DISPONIBLE'; and another 'SENSOR' by 'WATCHING' with model 12421, having tasks 'REVISION DE CONECTORES' and 'DISPONIBLE'. At the bottom are buttons for 'Eliminar', 'Guardar Orden', and a link to 'Instituto Nacional de Meteorología e Hidrología (INAMHI) | Derechos Reservados 2016 - Desarrollo de Sistemas'.

**Figura 87. Orden de trabajo llena**

En la opción Seguimiento de Órdenes se despliega una lista que nos muestra todas las órdenes existentes, se puede seleccionar una orden dando clic sobre la misma.

The screenshot shows a list of work orders. The sidebar menu under 'Gestionar Órdenes de Trabajo' includes 'Ingresar', 'Seguimiento de Órdenes', 'Mantenimiento', 'Área De Órdenes', 'Estado De Órdenes', and 'Tarea De Órdenes'. The main area is titled 'Seguimiento de Órdenes' and displays a table with columns 'ID', 'Área Orden', 'area solicitante', 'Nombre Solicitante', 'Fecha Creación', 'Fecha Entrega', and 'Estado Orden'. The table lists several orders: 'ELECT\_2016-06-03 00:00:00' (Electrónica, Desarrollo, Diego, 2016-06-03, 2016-06-23, Terminada); 'ELECT\_2016-06-09 00:00:00' (Electrónica, idt, bbb, 2016-06-09, 2016-06-10, Terminada); 'MEC\_2016-06-03 00:00:00' (Mecánica, Desarrollo, Diego, 2016-06-03, 2016-06-30, Solicitada); 'MET\_2016-05-01 00:00:00' (Metrología, IDT, Diego, 2016-05-01, 2016-05-24, Terminada); 'MET\_2016-05-26 00:00:00' (Metrología, Sistemas, Jorge, 2016-05-26, 2016-05-28, Cancelada); 'MET\_2016-06-03 00:00:00' (Metrología, Desarrollo, Diego, 2016-06-03, 2016-06-30, Solicitada); 'MET\_2016-06-08 00:00:00' (Metrología, idt, jake, 2016-06-08, 2016-06-09, Solicitada); and 'MET\_2016-06-22 00:00:00' (Metrología, Desarrollo, javi, 2016-06-22, 2016-06-25, Solicitada). At the bottom are buttons for 'Mostrar Detalle', 'Cerrar Orden', 'Cancelar Orden', and 'En Ejecución'.

**Figura 88. Lista de órdenes de trabajo**

Al dar clic sobre el botón Mostrar Detalle se abrirá una lista con los elementos vinculados a dicha orden.

The screenshot shows a web-based application for managing work orders. On the left, there is a sidebar with navigation links: 'Ingresar', 'Seguimiento de Órdenes', 'Mantenimiento', 'Area De Órdenes', 'Estado De Órdenes', and 'Tarea De Órdenes'. The main content area displays a table of work orders with columns: ID, Fecha, Elemento, Desarrollo, Asignado A, Fecha Creación, Fecha Finalización, and Estado. One row is selected, highlighted in yellow. Below the table is a toolbar with buttons: 'Mostrar Detalle' (highlighted in green), 'Cerrar Orden' (red button), 'Cancelar Orden' (white button with red X), and 'En Ejecución' (brown button). At the bottom, there is another table showing details for the selected order, with columns: ID Detalle Orden, ID Elemento, Marca, Línea, Nombre Elemento, and Estado Elemento. The data in both tables is as follows:

ID Detalle Orden	ID Elemento	Marca	Línea	Nombre Elemento	Estado Elemento
1	11	LOGOTRONIC	DATALOGER	4	EN METROLOGIA
2	11	LOGOTRONIC	DATALOGER	4	EN METROLOGIA
3	21	VAISALA	SENSOR	1	EN METROLOGIA
4	21	VAISALA	SENSOR	1	EN METROLOGIA
5	24	WATCHING	SENSOR	1	DISPONIBLE

**Figura 89. Elementos vinculados a orden de trabajo seleccionada**

El botón Cerrar Orden dará por finalizada poniéndola en estado “terminada” la orden seleccionada y hará disponibles sus respectivos elementos para que puedan ser utilizados nuevamente por otros empleados del INAMHI.

El botón Cancelar Orden pondrá el estado de la orden en “cancelada” y pondrá en “disponible” el estado de sus elementos.

El botón En Ejecución cambiará el estado de la orden a “En ejecución”. Todo esto se hace para saber dónde y en qué estado se encuentran los elementos, las solicitudes de elementos y las órdenes de trabajo de elementos.

## 4.2 PRUEBAS

A continuación se listan las pruebas que se realizaron sobre el aplicativo web:

**Tabla 6.**

### Caso de Prueba CP\_F1.5

<b>Nombre</b>	Caso de Prueba Ingreso al Sistema			
<b>ID Caso de Prueba</b>	CP_F1.5			
<b>Responsable Caso de Prueba:</b>	Jorge Escobar			
<b>Fecha Def Caso de Prueba:</b>	13-06-2016			
<b>ID Requerimiento:</b>	F1.5			
<b>Nombre Requerimiento:</b>	Ingreso al Sistema			
<b>Responsable Requerimiento:</b>	Jorge Escobar			
<b>ALCANCE</b>				
Definir la configuración de la transacción "F1.5" " Ingreso al Sistema ", para el proceso de pruebas en la autenticación de usuarios para el correcto ingreso al sistema				
<b>PREREQUISITOS</b>				
Ninguno				
<b>DATOS GENERALES DE PRUEBA</b>				
Tipo de Error: "Usuario/Contraseña inválidos"				
Tipo de Error: "El usuario no se encuentra registrado en el sistema"				
<b>CASOS DE PRUEBA</b>				
Orden	Descripción	Resultado esperado		
1	Ingresar el usuario			
2	Ingresar la contraseña			
3	Ingresar al sistema	Ingreso exitoso al sistema		
<b>RESULTADOS OBTENIDOS</b>				
Fecha	Resultado Obtenido			
15-06-2016	Ingreso exitoso al sistema			
20-06-2016	Ingreso exitoso al sistema			
23-06-2016	Ingreso exitoso al sistema			
<b>APROBACIÓN</b>				
<b>Aprobado por:</b>	José Sancho			
<b>Fecha Aprobación:</b>	28-06-2016			

**Tabla 7.**

### Caso de Prueba CP\_F1.3

<b>Nombre</b>	Caso de Prueba Actualizar Usuario	
<b>ID Caso de Prueba</b>	CP_F1.3	
<b>Responsable Caso de Prueba:</b>	Jorge Escobar	
<b>Fecha Def Caso de Prueba:</b>	13/06/2016	
<b>ID Requerimiento:</b>	F1.3	
<b>Nombre Requerimiento:</b>	Edición Usuario	
<b>Responsable Requerimiento:</b>	Jorge Escobar	
<b>ALCANCE</b>		

Definir la configuración de la transacción "F1.3" "Actualizar Usuario", para el proceso de pruebas en la modificación de usuario

#### **PREREQUISITOS**

Ejecutar el caso de prueba "CP\_F1.3"

#### **DATOS GENERALES DE PRUEBA**

Tipo de Error: "El usuario no tiene rol de administrador"

#### **CASOS DE PRUEBA**

<b>Orden</b>	<b>Descripción</b>	<b>Resultado esperado</b>
1	Entrar a Gestión de Usuarios	
2	Seleccionar un Usuario	
3	Edición de la propiedad de Edición exitosa en el sistema usuario que se desea modificar	

#### **RESULTADOS OBTENIDOS**

<b>Fecha</b>	<b>Resultado Obtenido</b>
15-06-2016	Edición exitosa del usuario
20-06-2016	Edición exitosa del usuario
23-06-2016	Edición exitosa del usuario

<b>APROBACIÓN</b>	
<b>Aprobado por:</b>	José Sancho
<b>Fecha Aprobación:</b>	28-06-2016

**Tabla 8.**

#### **Caso de Prueba CP\_F3.3**

<b>Nombre</b>	Caso de Prueba Actualizar Elemento
<b>ID Caso de Prueba</b>	CP_F3.3
<b>Responsable Caso de Prueba:</b>	Jorge Escobar
<b>Fecha Def Caso de Prueba:</b>	13/06/2016
<b>ID Requerimiento:</b>	F3.3
<b>Nombre Requerimiento:</b>	Edición de Elemento
<b>Responsable Requerimiento:</b>	Jorge Escobar

#### **ALCANCE**

Definir la configuración de la transacción " F3.3" "Actualizar Elemento", para el proceso de pruebas en la modificación de elemento

#### **PREREQUISITOS**

Ejecutar el caso de prueba "CP\_F3.3"

#### **DATOS GENERALES DE PRUEBA**

Tipo de Error: "El usuario no tiene rol de administrador"

#### **CASOS DE PRUEBA**

<b>Orden</b>	<b>Descripción</b>	<b>Resultado esperado</b>
1	Entrar a Gestión de Elementos	
2	Seleccionar un Elemento	
3	Edición de la propiedad de Edición exitosa en el sistema elemento que se desea modificar	

#### **RESULTADOS OBTENIDOS**

<b>Fecha</b>	<b>Resultado Obtenido</b>
15-06-2016	Edición exitosa del elemento

20-06-2016	Edición exitosa del elemento
23-06-2016	Edición exitosa del elemento
<b>APROBACIÓN</b>	
<b>Aprobado por:</b>	José Sancho
<b>Fecha Aprobación:</b>	28-06-2016

**Tabla 9.****Caso de Prueba CP\_F4.1**

<b>Nombre</b>	Caso de Prueba Ingresar Solicitud			
<b>ID Caso de Prueba</b>	CP_F4.1			
<b>Responsable Caso de Prueba:</b>	Jorge Escobar			
<b>Fecha Def Caso de Prueba:</b>	13/06/2016			
<b>ID Requerimiento:</b>	F4.1			
<b>Nombre Requerimiento:</b>	Ingresar Solicitud			
<b>Responsable Requerimiento:</b>	Jorge Escobar			
<b>ALCANCE</b>				
Definir la configuración de la transacción " F4.1" "Ingresar Solicitud", para el proceso de pruebas en la creación de solicitudes				
<b>PREREQUISITOS</b>				
Ejecutar el caso de prueba "CP_F4.1"				
<b>DATOS GENERALES DE PRUEBA</b>				
Tipo de Error: "El usuario no tiene rol de administrador"				
<b>CASOS DE PRUEBA</b>				
Orden	Descripción	Resultado esperado		
1	Entrar a Gestión de Solicitudes de Elementos			
2	Entrar a Ingresar Elemento			
3	Ingresar la estación para la cual se solicita el elemento			
4	Ingresar el nombre del solicitante			
5	Ingresar la fecha de solicitud			
6	Ingresar la fecha de salida a comisión			
7	Ingresar el motivo			
8	Añadir los elementos que se desea solicitar	Visualización en pantalla de lista del elementos a solicitarse		
9	Ingreso de solicitud de elemento	Ingreso exitoso en el sistema		
<b>RESULTADOS OBTENIDOS</b>				
Fecha	<b>Resultado Obtenido</b>			
15-06-2016	Ingreso exitosa del elemento			
20-06-2016	Ingreso exitosa del elemento			
23-06-2016	Ingreso exitosa del elemento			
<b>APROBACIÓN</b>				
<b>Aprobado por:</b>	José Sancho			
<b>Fecha Aprobación:</b>	28-06-2016			

**Tabla 10.****Caso de Prueba CP\_RF4.2.3**

<b>Nombre</b>	Caso de Prueba Aprobar parcialmente solicitud			
<b>ID Caso de Prueba</b>	CP_RF4.2.3			
<b>Responsable Caso de Prueba:</b>	Jorge Escobar			
<b>Fecha Def Caso de Prueba:</b>	13/06/2016			
<b>ID Requerimiento:</b>	RF4.2.3			
<b>Nombre Requerimiento:</b>	Aprobar parcialmente solicitud			
<b>Responsable Requerimiento:</b>	Jorge Escobar			
<b>ALCANCE</b>				
Definir la configuración de la transacción "RF4.2.3" "Aprobar parcialmente solicitud", para el proceso de pruebas en la creación de solicitudes				
<b>PREREQUISITOS</b>				
Ejecutar el caso de prueba "CP_RF4.2.3"				
<b>DATOS GENERALES DE PRUEBA</b>				
Tipo de Error: "El usuario no tiene rol de administrador"				
<b>CASOS DE PRUEBA</b>				
Orden	Descripción	Resultado esperado		
1	Entrar a Gestión de Solicituds de Elementos			
2	Entrar a Asignar Estado			
3	Seleccionar la solicitud que se quiere aprobar parcialmente			
4	Dar clic en mostrar detalle	Visualización en pantalla de una lista con los elementos vinculados a la solicitud seleccionada		
5	Seleccionar el o los elementos que se desean aprobar			
6	Aprobación parcial de la solicitud	Aprobación parcial exitosa en el sistema		
<b>RESULTADOS OBTENIDOS</b>				
Fecha	<b>Resultado Obtenido</b>			
15-06-2016	Ingreso exitosa del elemento			
20-06-2016	Ingreso exitosa del elemento			
23-06-2016	Ingreso exitosa del elemento			
<b>APROBACIÓN</b>				
<b>Aprobado por:</b>	José Sancho			
<b>Fecha Aprobación:</b>	28-06-2016			

**4.3 PLAN DE ENTREGA DE SOFTWARE****4.3.1 INTRODUCCIÓN**

El propósito de la definición del plan de entrega de software es establecer las actividades a realizarse al momento de entregar el producto, tomando en cuenta no solo el software sino también la documentación asociada al proyecto. Se requiere hacer las

entregas comunes entre si para hacer más fácil su revisión y tratamiento. Este proceso llevará desde la formalización de la entrega hasta la revisión y validación de la misma.

#### **4.3.2 ALCANCE**

El plan de entrega de software durará el tiempo que duren las actividades que lo componen, las cuales se irán realizando una tras de otra. Dichas actividades vinculan el software y la documentación del proyecto con los recursos humanos involucrados coordinando una planificación organizada, la cual permite una visión detallada del procedimiento de entrega del sistema web desarrollado.

#### **4.3.3 PROCEDIMIENTO DE ENTREGA DE SOFTWARE**

El procedimiento de entrega de software establece el grupo de actividades que se requieren para proceder a la entrega del mismo y de la documentación vinculada al proyecto, este proceso tiene un alcance que incluye la formalización, la revisión y la validación de la entrega del producto.

Las actividades de entrega de software son:

- i. Entrega de producto software.
- ii. Entrega de documentación.
- iii. Comprobación del producto y revisión de la documentación.
- iv. Rechazo de la entrega.
- v. Ejecución de los servicios de testing.
- vi. Validación de la entrega.
- vii. Aceptación de la entrega.
- viii. Actualización de incidencias resueltas con la entrega.
- ix. Capacitación

## CAPÍTULO 5

### CONCLUSIONES Y RECOMENDACIONES

#### 5.1 Conclusiones

- Los artefactos RUP sirven de guía para que el desarrollador tenga una idea concreta de lo que el sistema tiene que hacer y cómo se va a llegar a ese funcionamiento. Hay aspectos importantes como la construcción de los diagramas de casos de uso, clases, despliegue, etcétera, los cuales son una referencia para construir los módulos del sistema. A pesar de esto muchas cosas pueden ir cambiando mientras se va construyendo el sistema con el objetivo de hacerlo más amigable con el usuario.
- Durante el desarrollo se utilizaron varias herramientas para las distintas capas, el desarrollador puede escoger las mismas dependiendo la medida en que éstas le ayuden en su trabajo. Se pudo usar varias herramientas que facilitaron y ahorraron tiempo y esfuerzo en el desarrollo como son Hibernate, Spring y JSF. Estas herramientas son muy flexibles y pueden ser configuradas de acuerdo a las necesidades del programador.
- Spring permite ahorrar muchas líneas de código al otorgar la posibilidad de crear roles a los cuales se les puede asignar a qué página pueden ingresar. Esta es una manera rápida de controlar que usuarios no puedan ver o modificar información que no les corresponde.
- Fue imprescindible hacer un bean genérico el cual fue extendido por todos los demás beans para manejar el uso de funciones comunes como guardar o borrar un objeto nuevo o uno ya existente, así como el manejo de objetos seleccionados en cada página web.
- Para poder tener escalabilidad en el sistema se necesitó mantener las capas independientes las unas de las otras, así cuando fue necesario realizar cambios se pudo realizarlos solamente en la capa pertinente sin afectar al funcionamiento del resto del proyecto. Esto hizo que el sistema sea mucho más flexible y fácil de mantener. El sistema podrá seguir aumentando módulos a futuro sin afectar su funcionamiento actual.
- En el desarrollo de una aplicación web se puede utilizar solamente los módulos de Spring que se necesita, más no toda la herramienta en su totalidad, demostrando así su alta cohesión. Este framework puede también integrarse con

otras herramientas dependiendo de las necesidades del usuario.

- Una consecuencia de la implementación del sistema fue la optimización y automatización de los procesos de gestión de elementos que posee el INAMHI.
- Con una constante lectura y estudio se logró utilizar las tecnologías JEE con Spring y la metodología RUP para la construcción del sistema.
- Se Implementó el sistema en el servidor de la empresa y se implantó el mismo en las instalaciones del INAMHI siendo sus procesos ahora normas que tienen que seguir los empleados de la organización.

## 5.2 Recomendaciones

- Reducir al mínimo posible las dependencias entre capas para que cuando se realice cambios o actualizaciones no afecte al funcionamiento de la aplicación.
- Utilizar un bean para cada objeto, de esta forma se puede manipular de manera independiente las variables que toman diferentes valores en una página web.
- Si se desea mejorar la vista de una aplicación existen plantillas hmtl5 en internet que vienen con sus propios css desde gratuitas hasta pagadas los cuales añadiendo las partes que se requieren a la capa vista pueden dar una apariencia más profesional a una aplicación web.
- Se recomienda priorizar una estructura de bajo acoplamiento desde el principio hasta el final de la construcción de un sistema web para poder realizar cambios o actualizaciones a futuro por parte del departamento de desarrollo después de que el producto sea entregado.
- Para el desarrollo de aplicaciones web usar el IDE Netbeans ya que además de ser gratis es amigable con el usuario, facilita la gestión de proyectos grandes y además el depurado de las aplicaciones es sencillo.

## REFERENCIAS BIBLIOGRAFIAS

- Apache Maven Organization (2016). What is Maven. [En línea]. Recuperado de: <https://maven.apache.org/what-is-maven.html#>
- Ceballos, F.J.(2003). *Java 2 Curso de programación*, México, D.F.:Pitágoras.
- Chandi, L, & Roldan, G. (2015). *Implementación de un aplicativo web como servicio SAAS, bajo una infraestructura en la nube IAAS, para la cooperativa San Vicente del Sur-matriz* (tesis de pregrado). Universidad de las Fuerzas Armadas “ESPE”, Ecuador.
- Deitel, P., & Deitel, H. (2012). *Java: Como programar*. México, D.F.: Pearson Educación.
- Fernandez, J, & Cadelli, S. (2014). *Convivencia de metodologías: Scrum y Rup en un proyecto de gran escala* (tesina de licenciatura). Universidad Nacional de la Plata, Argentina.
- Firtman, M.(2010). *Ajax: web 2.0 con jQuery para profesionales*. Buenos Aires, Argentina: Alfaomega.
- Flores, J, & Puruncajas, R. (2015). *Análisis, diseño e implementación de un sistema web que permita automatizar el procedimiento de titulación en posgrado de la Universidad de las Fuerzas Armadas “ESPE”, mediante la utilización de la plataforma java Enterprise edition JEE6 web aplicando la metodología SCRUM* (tesis de pregrado). Universidad de las Fuerzas Armadas “ESPE”, Ecuador.
- Fowler, M., “Plain Old Java Object (POJO)”, Recuperado de: <http://www.martinfowler.com/bliki/POJO.html> (accedido 07/06/2013).
- Johnson, R. (2005). *Professional Java Development with the Spring Framework*. Pennsylvania, USA: Wiley Publishing Inc.
- Larman, C.(2004). *UML y Patrones*. Murcia, España: Pearson Educación.
- Oracle. (2010). *The Java EE 5 Tutorial*. Recuperado de <https://docs.oracle.com/javaee/5/tutorial/doc/bnaaw.html>

- Oracle Corporation (2016). Java Enterprise Edition (Java EE). [En línea]. Recuperado de:  
<https://www.java.com/es/download/faq/techinfo.xml>
- Romero, G.(2004). *UML con Rational Rose*. Cuenca, Ecuador: Megabyte.
- Suárez, D. (2011). *Spring framework Desarrollo JEE ágil*. Recuperado de  
<http://sg.com.mx/content/view/361>
- Vivas, L., & García, N., & Petroff, M & Abbate, H. (2013). *Un Marco de Trabajo para la Integración de Arquitecturas de Software con Metodologías Ágiles de Desarrollo* (objeto de conferencia). Universidad Nacional de la Plata, La Plata, Argentina.