

**ADAPTACIÓN DE UN SISTEMA DE TELECITOLOGÍA A UNA ARQUITECTURA
MODULAR MEDIANTE LA UTILIZACIÓN DEL FRAMEWORK OSGI**

DIANA CONSTANZA TARAZONA CARREÑO

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA INGENIERÍA DE SISTEMAS E INFORMATICA
BUCARAMANGA

2014

ADAPTACIÓN DE UN SISTEMA DE TELECITOLOGÍA A UNA ARQUITECTURA MODULAR MEDIANTE LA UTILIZACIÓN DEL FRAMEWORK OSGI

DIANA CONSTANZA TARAZONA CARREÑO

Trabajo de grado para optar por el título de
Ingeniera de Sistemas

Director

Master Fernando Antonio Rojas Morales
Docente de la Escuela de Ingeniería de Sistemas
Universidad Industrial de Santander

Tutor

Ing. HUGO ARMANDO GUALDRON COLMENARES
Desarrollador de Software
Biosys Ltda

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA

2014

A mi esposo Hugo Armando, mi hijo Santiago Alejandro, a mis Padres Gustavo Tarazona y Leonor Carreño, A mis hermanos Gustavo, Ivonne y Robinson

AGRADECIMIENTOS

La autora expresa sus inmensos agradecimientos a:

A Papito Dios y Mamita María por darme la sabiduría, entendimiento, fuerza y persistencia a lo largo de todos mis estudios y culminación de estos.

A mis Padres Gustavo Tarazona y Leonor Carreño por el amor, animo, fuerza y acompañamiento en los momentos difíciles de cada una de las etapas de mi aprendizaje. Porque sin ellos no sería lo que soy hoy en día. Gracias.

A mi esposo Hugo Armando por darme un inmenso acompañamiento y motivación durante todo mi proceso profesional y por ser mi modelo a seguir. Gracias amor por esa energía motivadora que crece en ti.

A ti mi hijo amado Santiago Alejandro, por ser mi motor de vida y de felicidad en mi diario vivir.

A mis hermanos Gustavo Adolfo, Ivonne Melina y Robinson Daniel porque siempre han estado en mis momentos de tristeza y felicidad.

A Juanita porque me abrió las puertas de su Institución para empezar a ejercer en mi vida laboral.

Al Profesor Alfonso Mendoza por creer y confiar en mí, y apoyarme en este proceso.

Al Profesor Fernando Rojas por sus enseñanzas en mi proceso como profesional. Gracias Profesor por el apoyo y ejemplo de muchos profesionales que hoy en día quieren seguir el camino de educar.

A Miguel y Melissa por su aporte en todas las correcciones de mi escritura y por su fuerza, compromiso y dedicación en Biosys Ltda.

A todos mil y mil gracias, este éxito lo dedico a ustedes.

CONTENIDO

	Pág.
INTRODUCCIÓN	15
1. DESCRIPCIÓN DE LA PRÁCTICA	17
1.1. JUSTIFICACIÓN	17
1.2. DEFINICIÓN DEL PROBLEMA	18
1.3. OBJETIVO GENERAL	19
1.4. OBJETIVOS ESPECÍFICOS	20
1.5. IMPACTO	20
1.6. METODOLOGÍA DE LA PRÁCTICA EMPRESARIAL	21
2. BIOSYS LTDA	23
2.1. MISIÓN	23
2.2. VISIÓN	23
3. MARCO TEÓRICO	24
3.1. OPEN SERVICES GATEWAY INICIATIVE OSGI	24
3.1.1 CARACTERÍSTICAS DE LA ESPECIFICACION OSGI	25
3.1.2 BENEFICIOS QUE PROPORCIONA ESTA TECNOLOGIA	26
3.2 ARQUITECTURA SOFTWARE DE OSGI	27
3.3 FRAMEWORK OSGI	27
3.3.1 CAPA MÓDULO	27
3.3.2 CAPA CICLO DE VIDA	30
3.3.3 CAPA DE SERVICIO	31

3.4. VAADIN	33
3.4.1. ARQUITECTURA DE VAADIN	33
4. DESARROLLO DE LA PRÁCTICA EMPRESARIAL	34
4.1. APLICACIONES EMPRESARIALES EN BIOSYS LTDA	34
4.2. CITOSOFT: SISTEMA DE TELECITOLOGIA PARA APOYO AL PROGRAMA DE DETECCIÓN TEMPRANA DE CÁNCER CÉRVICOUTERINO EN COLOMBIA	35
4.3. LIMITACIONES DE LA IMPLEMENTACION DE CITOSOFT	42
4.4. MODULARIZACION DE CITOSOFT EN OSGI	44
4.5. ARQUITECTURA BIOSYS LTDA	44
4.6. VAADIN Y OSGI	46
4.6.1. EXTENSION DE VAADIN	47
4.7. MODULOS DE CITOSOFT	52
4.8. ADMINISTRACION DE APLICACIONES EN BIOSYS LTDA	56
4.9. MODULOS ASOCIADOS AL PROCESO DE DIAGNOSTICO BASADO EN CITOLOGIA CERVICOUTERINA	56
4.10. PRUEBAS DE FUNCIONALIDAD DEL FRAMEWORK OSGI EN EL SISTEMA CITOSOFT	56
CONCLUSIONES	67
BIBLIOGRAFÍA	68
REFERENCIAS BIBLIOGRAFICAS	71

LISTA DE FIGURAS

	Pág.
Figura 1. Arquitectura en Capas del Framework OSGi	25
Figura 2. Bundle con su respectivo código, recursos y metadatos.	26
Figura 3. Descripción de un paquete exportado.	27
Figura 4. Descripción de un paquete importado.	28
Figura 5. Ciclo de Vida de un Bundle.	29
Figura 6. La interacción orientada a servicios. Proveedores de servicios publican en un registro donde los solicitantes pueden descubrir cuáles son los servicios disponibles para su uso.	30
Figura 7. Vista de usos de alto nivel de CITOSOFT.	34
Figura 8. Casos de uso para la fase de Pre análisis del proceso de diagnóstico basado Citología Cervicouterina.	35
Figura 9. Casos de uso para la fase de análisis del proceso de diagnóstico basado Citología Cervicouterina.	36
Figura 10. Casos de uso para la fase de Pos análisis del proceso de Citología Cervicouterina.	37
Figura 11. Ejemplo de interfaz dinámica con soporte a contenedores dinámicos.	45
Figura 12. Ejemplo de versionamiento de servicios UIService.	47
Figura 13. Soporte por defecto del Framework Vaadin a la Arquitectura OSGi.	49
Figura 14. Soporte del Framework de Vaadin extendido a la Arquitectura OSGi.	50
Figura 15. Diagrama de dependencias de los paquetes del núcleo de las aplicaciones Biosys.	51
Figura 16. Dependencia de los paquetes del núcleo de biosys después de la modularización.	53

LISTA DE TABLAS

	Pág
Tabla 1. Pruebas no funciones asociadas a la modularizacion.	57
Tabla 2. Pruebas realizadas a los requerimientos funcionales del Sistema Citosoft.	60

LISTADO DE SIGLAS

- AJAX: Asynchronous Java Script and XML
- API: Application Programming Interface.
- AWT: Abstract Window Toolkit.
- EAR: Enterprise ARchive
- EPS: Entidad Promotora de Salud
- GWT: Google Web Toolkit.
- HTML: HyperText Markup Language
- IPS: Institución Prestadora de Servicios
- JAR: Java ARchive.
- JDBC: Java Database Connectivity
- JDK: Java Development Kit
- JEE: *Java Platform, Enterprise Edition*
- JEP: Java Enterprise Process
- JMX: Java Management eXtensions
- JNLP: Java Network Launching Protocol
- JPA: Java Persistence API
- JSE: Java Standard Edition
- JTA: Java Transaction API
- OSGi: OPEN SERVICES GATEWAY INICIATIVE
- RIA: Aplicaciones de Internet Enriquecidas.
- SWT: Standard Widget Toolkit.
- TIC: Tecnologías de la Información y la Comunicación.
- UI: Interfaz de Usuario
- XML: Extensible Markup Language.
- WAR: Web Application Archive
- WEB: World Wide Web

RESUMEN

TÍTULO: ADAPTACIÓN DE UN SISTEMA DE TELECITOLOGÍA A UNA ARQUITECTURA MODULAR MEDIANTE LA UTILIZACIÓN DEL FRAMEWORK OSGI

AUTOR: TARAZONA CARREÑO, Diana Constanza^{1**}

PALABRAS CLAVES: Framerwork Vaadin, Telecitologia, Cáncer, Citología, Arquitectura Modular

DESCRIPCION: A pesar de los esfuerzos de detección temprana de cáncer, la baja eficiencia de la tamización por citología de cérvix uterino (Papanicolaou) ha derivado en una alta incidencia de este tipo de cáncer en Colombia. Anualmente mueren cerca de 2000 mujeres y son detectados 5.500 casos nuevos, siendo su segunda causa de muerte por cáncer. Uno de los factores que ha acrecentado este problema es la falta de sistemas de información que apoyen el proceso de diagnóstico basado en Citología cervicouterina. Para contrarrestar esta problemática, Biosys Ltda. desarrolló CITOSOFT, un Sistema de Telecitología de apoyo al programa de detección temprana de cáncer cérvicouterino en Colombia con el financiamiento del Departamento Administrativo Ciencia, Tecnología e Innovación COLCIENCIAS, que actualmente se encuentra en segunda fase. Con el objetivo de prevenir potenciales complicaciones en su fase de implantación, se propuso la adopción de una arquitectura modular soportada por el Framework OSGi que integra las características del Framework Vaadin, simplificando la fase de mantenimiento de este software, permitiendo en tiempo de ejecución actualizar componentes individuales sin afectar el funcionamiento global del sistema; trabajo que se realizó en el marco de una práctica empresarial para optar el título de ingeniera de sistemas por la autora de ese documento.

*Trabajo de Grado

** Facultad de Ingenierías Físico Mecánicas. Escuela de Ingeniería de Sistemas e Informática.
Director de Proyecto: Master Fernando Antonio Rojas Morales. Tutor: Ing. Hugo Armando Gualdrón Colmenares

ABSTRACT

TITLE: ADAPTATION FROM A TELECYTOLOGY SYSTEM TO A MODULAR ARCHITECTURE BY USING THE FRAMEWORK OSGI

AUTHOR: TARAZONA CARREÑO, Diana Constanza **

KEYWORDS: Vaadin Framework, Telecytology, Cancer, Cytology, OSGI Framework.

DESCRIPTION: Despite the efforts of early cancer detection, the low efficiency of screening in uterine cervix cytology (Papanicolaou technique) has resulted in a high incidence of this cancer in Colombia. One factor that has increased the problem is the lack of information systems that support the process of diagnosis based on cervical cytology. To face this problem, the company BIOSYS LTDA, developed a Telecytology system called CITOSOFT to support the early detection of cervical cancer in Colombia. In order to prevent potential complications in its implementation phase, adopting a modular architecture supported by the OSGi Framework that integrates the features of the Vaadin Framework to simplify the maintenance phase of the software is proposed, allowing upgrade individual components without affecting the overall operation of the system; this work was carried out within the framework of a business practice to get the title of systems engineer by the author of this document.

* Minor Degree Thesis

** Faculty of Physical and Mechanical Engineering. School of System Engineering. Project Director: Master. Fernando Antonio Rojas Morales . Tutor: Eng. Hugo Armando Gualdron Colmenares

INTRODUCCIÓN

El cáncer de cuello uterino es el tercer tipo de cáncer más frecuente en mujeres a nivel mundial. Este tipo de cáncer es una enfermedad responsable de 275.000 muertes en el 2008, donde el 88% de tasas de incidencia ocurrieron en países en desarrollo como África, América Central y América del Sur, el Caribe y en Asia.[18]

En Colombia continúa siendo la segunda causa de muerte por cáncer en mujeres, a pesar de la implementación desde 1991 de la detección temprana basada en la citología cervicouterina o prueba de Papanicolaou. Según el Atlas de mortalidad por cáncer en Colombia, el patrón de riesgo de mortalidad presenta muchas áreas del territorio nacional con riesgos incrementados; particularmente las zonas aledañas a los ríos navegables, las áreas de frontera, el golfo de Urabá y la serranía del Baudó.[19]

En países desarrollados la implementación de programas de tamización para cáncer de cuello uterino mediante la citología cervicouterina ha mostrado efectividad en la reducción del 80% de la mortalidad y la incidencia de cáncer de cuello uterino en países en desarrollo como Colombia no se ha logrado un mayor descenso en la mortalidad. Las razones a esta caída son la falta de efectividad en la tamización en las naciones de bajos recursos, además de las limitaciones socioeconómicas, los problemas en la organización de los programas de salud como lo es: la calidad en la toma y en la lectura de la citología, el acceso oportuno al diagnóstico y al tratamiento, a la calidad del tratamiento y a la insuficiencia de los sistemas de información que le dan soporte a este proceso. [20]

Actualmente las entidades prestadoras del servicio de salud se encuentran en vías a mejorar en términos de calidad, eficiencia y eficacia la prestación de servicios

por medio de la Telemedicina. Por esto se hace necesario que la información que manejan se encuentre disponible y de fácil acceso, teniendo en cuenta normas de seguridad y privacidad.

El desarrollo de esta práctica empresarial busca mejorar el Sistema de Telecitología de apoyo al programa de detección temprana de Cáncer Cervicouterino CITOSOFT de Biosys Ltda., extendiendo el Framework de desarrollo de aplicaciones web Vaadin a la arquitectura modular de OSGi, mejorando la capacidad de respuesta de la empresa a situaciones que impliquen inclusión o corrección de procesos.

El documento comienza con la descripción de la práctica empresarial realizada. Posteriormente se presenta a la empresa Biosys Ltda donde se realizó la práctica empresarial, luego se enfoca al lector en la temática abordando los conceptos referentes a la especificación OSGi, su arquitectura, así como las tecnologías utilizadas en la organización. Posteriormente se explica el desarrollo de la práctica empresarial y finalmente se presentan las conclusiones.

1. DESCRIPCIÓN DE LA PRÁCTICA

1.1. JUSTIFICACIÓN

La Práctica Empresarial en Biosys, no sólo permite a la estudiante aplicar sus conocimientos y habilidades adquiridas a lo largo de la carrera universitaria; sino también ayuda en su desarrollo profesional y laboral, preparándola para enfrentar los desafíos del mundo al que tendrá que afrontar luego de obtener su título como profesional.

La Escuela de Ingeniería de Sistemas e Informática - EISI se verá beneficiada con este proyecto en la medida en que se promueve el futuro profesional de la estudiante, el vínculo empresarial, el desarrollo de un perfil profesional propio basado en las características de una Ingeniera de Sistemas UIS.

Para la empresa, la practicante será un apoyo en la implementación de la arquitectura OSGi en el Sistema de Telecitología de apoyo al programa de detección temprana de cáncer cérvicouterino CITOSOFT, que permitirá modularizar e incrementar la escalabilidad del sistema, mejorando la capacidad de respuesta de Biosys Ltda. a situaciones que impliquen inserción o corrección de procesos.

Es importante destacar que realizar la práctica empresarial en una organización como lo es Biosys, que con esfuerzo, dedicación y ayuda de jóvenes emprendedores y practicantes está desarrollando su proyecto tecnológico en el sector de la salud para darse a conocer a nivel Departamental y Nacional; es una gran oportunidad para la autora de este proyecto y el buen nombre de la Escuela de Ingeniería de Sistemas.

1.2. DEFINICIÓN DEL PROBLEMA

El cáncer cérvico uterino o cáncer de cuello uterino, es una enfermedad en la cual se encuentra células cancerosas (malignas) en los tejidos del cuello uterino. Es una clase común de cáncer en la mujer. Por lo general el cáncer de cuello uterino empieza creciendo lentamente. Antes que aparezcan células cancerosas en el cuello uterino, los tejidos normales de este pasan a experimentar cambios y empiezan a aparecer células anormales (este proceso es conocido como displasia); después, las células cancerígenas comienzan a crecer y se diseminan con mayor profundidad en el cuello uterino y en las áreas circundantes [1].

Este tipo de cáncer en Colombia es la segunda causa de muerte por cáncer femenino; donde cada año mueren cerca de 2.000 mujeres y se esperan 5.500 casos nuevos. De cada cien mujeres colombianas, más de dos lo padecerán antes de llegar a los 65 años de edad [2]. Estas cifras alarman al sistema de salud, ya que es prevenible y curable si se detecta a tiempo.

Biosys Ltda. con el apoyo del Departamento Administrativo Ciencia, Tecnología e Innovación COLCIENCIAS, ha desarrollado un sistema denominado CITOSOFT. Está es una solución informática que permite el manejo de información médica digital como imágenes celulares e historia clínica, generación de indicadores epidemiológicos y de morbi mortalidad de las enfermedades cérvicouterinas, indispensables para tomar decisiones sobre el programa de detección temprana de cáncer de cuello uterino en países en desarrollo donde la enfermedad se presenta con mayor incidencia [3].

Mediante el uso de este sistema, se evita el traslado físico de las muestras y de la información de las pacientes, brindando herramientas interactivas a los médicos patólogos y citotecnólogos a la hora de hacer el diagnóstico, permitiéndoles

observar con más detalle las características morfológicas de las células cervicales; Además proporciona a las entidades de salud información sobre el estado de sus pacientes, manejo de historias clínicas y la posibilidad a los municipios de conocer la incidencia de enfermedades, entre otros.

Actualmente CITOSOFT es un sistema que sólo puede ser desplegado en una única instancia de tiempo, y si se requiere mejorar la funcionalidad de algún proceso o adicionar uno nuevo al funcionamiento actual del sistema, es necesario notificar a todos los usuarios del sistema sobre la mejora, y efectuar una interrupción de las actividades de los diferentes especialistas que interactúan a través del sistema. A pesar de que estas interrupciones son programadas en tiempos de baja demanda de los servicios del sistema procurando afectar a la menor cantidad de usuarios y fallas críticas que no pueden esperar la programación del despliegue de una nueva versión deben ser ejecutadas en la mayor brevedad posible, ocasionando una interrupción del servicio de forma generalizada. Como solución a situaciones de esta naturaleza, en la presente práctica empresarial se pretende efectuar una modularización del software CITOSOFT, empleando el Framework OSGi, permitiendo así disponer de mecanismos que brindan gran flexibilidad en la configuración de las aplicaciones, que mediante detección temprana de dependencias ayuden a evitar errores en tiempo de ejecución y un control de versiones modular de las aplicaciones.

1.3. OBJETIVO GENERAL

Adaptar el Sistema de Telecitología de apoyo al programa de detección temprana de cáncer cérvicouterino (CITOSOFT) de la empresa Biosys Ltda. a una arquitectura modular mediante la utilización del Framework OSGi.

1.4. OBJETIVOS ESPECÍFICOS

- Definir los niveles de la arquitectura OSGi a implementar basado en los servicios y procesos que ofrece CITOSOFT.
- Implementar en cada uno de los servicios de CITOSOFT, las interfaces y clases abstractas derivadas del Framework OSGi, que garanticen un ciclo de vida dinámico de la aplicación.
- Elaborar la documentación acerca del diseño e implementación de módulos sobre el Framework OSGi, direccionada a las herramientas de desarrollo utilizadas en Biosys Ltda.
- Realizar pruebas de funcionalidad que permitan comprobar las ventajas de utilizar el Framework OSGi en el desarrollo de aplicaciones empresariales sobre la plataforma JAVA Enterprise Edition JEE.

1.5. IMPACTO

La adaptación de CITOSOFT a la arquitectura del Framework OSGi, mejorará la calidad de la prestación de servicios de telemedicina, proporcionando un sistema robusto, escalable y adaptable.

Adicionalmente esta mejora en el sistema CITOSOFT, contribuye a los objetivos del Plan Nacional de Desarrollo 2010-2014, el cual busca mejorar el estado de salud de la población en cuanto a eficiencia, calidad, equidad y sostenibilidad de los servicios; mediante el uso de Tecnologías de Información y Comunicación TIC,

procurando fortalecer la política de Telesalud en sus cuatro componentes: Telemedicina, Teleeducación, Epidemiología, Gestión y Administración.

Se espera un impacto positivo para Biosys, pues es de gran importancia el aporte del Framework OSGi para la empresa y sobre todo para CITOSOFT porque permite la catalogación y ejecución de componentes de software resolviendo a su vez problemas de mantenibilidad, flexibilidad, modularidad y reutilización, verificando que por medio de servicios se separe claramente la interfaz de los componentes de su correspondiente implementación, por ello es un logro muy importante.

Este efecto positivo no solo será de gran importancia para Biosys, sino también para la Escuela de Ingeniería de Sistemas e informática (EISI), la cual se verá beneficiada por desarrollo de este importante trabajo, así como sus aspectos más esenciales del uso del Framework OSGi y su arquitectura en el desarrollo de software.

El proyecto es viable ya que en la empresa se dispone de gran conocimiento en la plataforma Java Standard Edition (Java SE) y Java Enterprise Edition (Java EE), así como del uso de Frameworks de desarrollo Web como Vaadin, Struts, Hibernate, GWT, entre otros. A partir de este conocimiento y mediante capacitaciones a la estudiante en el uso de las tecnologías involucradas en el desarrollo de CITOSOFT, es posible llevar a cabo el proyecto en el tiempo propuesto.

1.6. METODOLOGÍA DE LA PRÁCTICA EMPRESARIAL

La metodología utilizada para el desarrollo de la práctica empresarial corresponde a la de un trabajo investigativo, analítico y práctico con estudio previo del sistema

CITOSOFT y todo lo referente a la nueva mejora de su arquitectura por medio del Framework OSGi.

Los pasos para desarrollar la metodología inicialmente mencionada, se explicarán por medio de una serie de aspectos que se cumplieron a lo largo de la práctica en la empresa:

- a.** Estudio, revisión, conocimiento y entendimiento del Framework OSGi, sus especificaciones así como las implementaciones existentes.
- b.** Recopilación de la información y documentación de la arquitectura actual del sistema CITOSOFT y las tecnologías empleadas en su desarrollo.
- c.** Estudio del Framework Vaadin y certificación en las habilidades y conocimientos adecuados acerca de Vaadin Framework 7 y su diferentes herramientas, para el desarrollo exitoso de aplicaciones web.
- d.** Elaboración e implementación de cada uno de los servicios de CITOSOFT, las interfaces y clases abstractas derivadas del Framework OSGi, que garanticen un ciclo de vida dinámico de la aplicación.
- e.** Reuniones con el director del proyecto, el tutor y la practicante.
- f.** Se elaboraron dos reportes entregados al director del proyecto, y una reunión con el tutor, el director y la practicante.

2. BIOSYS LTDA

Biosys Ltda. es una empresa de Ingeniería Biomédica dedicada a la Investigación y Desarrollo (I+D) de Soluciones informáticas, que dan soporte al diagnóstico médico en la especialidad de Patología, a los programas de Promoción y Prevención, y al aprendizaje y capacitación tanto de personal médico como estudiantes de la especialidad de Patología.

Biosys Ltda. nace de la necesidad de soluciones informáticas que permitan aumentar la calidad en los servicios de salud de la región, y así contribuyan a un incremento en la cobertura de servicios especializados de poblaciones marginadas o geográficamente distantes, que no pueden ser ofrecidos por las Empresas Prestadoras de Servicios (EPS) con la infraestructura tecnológica actual.

2.1. Misión

Biosys es una Empresa dedicada a la Investigación, Diseño y Desarrollo de Soluciones Informáticas de la más alta calidad que tienen como fin dar soporte al sistema de salud buscando siempre contribuir al aumento en la calidad y en la cobertura de la prestación de servicios especializados de salud tanto en la región, como en el país[4].

2.2 Visión

En el año 2016 la empresa será reconocida a nivel regional por la excelencia en sus creaciones científicas y tecnológicas, por su participación en el desarrollo del sector salud y tecnológico en la región y en la contribución al mejoramiento en la calidad de vida de la población, además, estará expandiendo sus soluciones a nivel nacional [4].

3. MARCO TEÓRICO

Para que una empresa tenga éxito, se mantenga en el mercado y esté al nivel de sus competidores, debe adaptarse a los cambios producto de la globalización y las nuevas tecnologías para el desarrollo de aplicaciones software. A pesar de que estos son temas que han sido mencionados desde hace tiempo, muchas de las grandes empresas en Colombia aún no cuentan siquiera con una base de aplicación para desarrollo software que contenga una estructura modular y escalable como la proporcionada por tecnologías como el Open Service Gateway Initiative (OSGi). Por tanto, las empresas deben ser creativas, innovadoras y abiertas al cambio; pero no sólo esto, deben ser cautelosas respecto a las amenazas que pueden afectar el buen funcionamiento de la organización, y es por esto que OSGi se considera un elemento clave que impulsa la flexibilidad, comprensión, seguridad, usabilidad y reusabilidad en los proyectos de software de toda organización [5].

3.1. OPEN SERVICES GATEWAY INITIATIVE OSGi

OSGi (Open Service Gateway initiative) es una alianza de empresas tecnológicas a nivel mundial fue creada en el año de 1999, está compuesta por una plataforma basada netamente en JAVA cuyo objetivo principal de Open Services Gateway Initiative (OSGi) es definir especificaciones abiertas que permitan la implementación y diseño de plataformas compatibles que puedan proporcionar múltiples servicios.

Entre las empresas que componen este consorcio, podemos encontrar compañías de diversa índole: automoción, aeronáutica, fabricantes de electrodomésticos, telecomunicaciones, fabricantes de teléfonos. Algunos ejemplos de miembros:

Motorola, Nokia, Mitsubishi Electric Corporation, Vodafone Group Services, LinkedIn, LG Electronics, etc [6].

La alianza de empresas definió el Framework OSGi que proporciona una infraestructura eficiente para el desarrollo de aplicaciones basadas en servicios dentro de una máquina virtual JAVA (JVM)[5]. Los principales objetivos del Framework OSGi son dar un mejor control sobre el código de las aplicaciones y capacidad para gestionar dinámicamente su ciclo de vida por medio de unidades de modularización conocidas como *Bundles*, sin necesidad de reiniciar el contenedor. Esto hace que esta tecnología sea muy adecuada para aplicaciones móviles, dispositivos integrados y aplicaciones de servidores [5] [7].

3.1.1 CARACTERÍSTICAS DE LA ESPECIFICACIÓN OSGi.

Las características generales de la especificación OSGi son [8]:

- Visible: Permite la visibilidad del código por archivo JAR, permitiendo especificar qué paquetes son visibles externamente.
- Portable: Las APIs de OSGi pueden ser flexibles e implementadas en un gran rango de plataformas, haciendo posible la adaptación a múltiples soluciones en todos los ámbitos.
- Segura: Incorpora un modelo de seguridad que cubre varios niveles, permitiendo proporcionar un entorno de ejecución seguro.
- Estandarizable: Ofrece una plataforma común para todos los fabricantes de dispositivos, desarrolladores y proveedores de servicios para facilitar el avance de la tecnología.
- Abierta: No obliga el uso de ninguna tecnología determinada. Permite diseñar plataformas que puedan proporcionar múltiples servicios. Solo se

deben desarrollar aplicaciones y servicios compatibles con las APIs independientes y definidas.

- Escalable: Permite a los proveedores personalizar y administrar fácilmente la plataforma.
- Dinámica: Es posible interactuar dinámicamente con el ciclo de vida de la aplicación, ya que los *Bundles* pueden ser instalados, actualizados, detenidos, inicializados o desinstalados en cualquier momento, permitiéndole a la aplicación adaptarse a las necesidades específicas del usuario.
- Modular: Permite que el código de una aplicación software se divida en partes lógicas que representan preocupaciones específicas. Esto hace que las aplicaciones hagan uso del Framework OSGi presentando tanto dependencia como interdependencia entre módulos totalmente controlada.
- Independiente al Hardware: La tecnología tiene que ser independiente al hardware, de forma que pueda funcionar con soluciones múltiples en el ámbito de los procesadores, los electrodomésticos, las soluciones domóticas, etc.
- Fiable: La plataforma debe tener buen desempeño de funcionamiento, sin caídas del sistema por descuidos o provocados malintencionadamente.

3.1.2 BENEFICIOS QUE PROPORCIONA ESTA TECNOLOGÍA

- Reutilización de código.
- Simplifica los proyectos en los que participan muchos desarrolladores en diferentes equipos.

- Se puede utilizar en sistemas más pequeños o embebidos.
- No es una tecnología cerrada [8].

Para la administración y desarrollo de aplicaciones modulares es necesaria la utilización de plataformas de servicios como OSGi, donde los servicios son encapsulados en componentes pueden ser administrados individualmente. A continuación se especificará la arquitectura de OSGi.

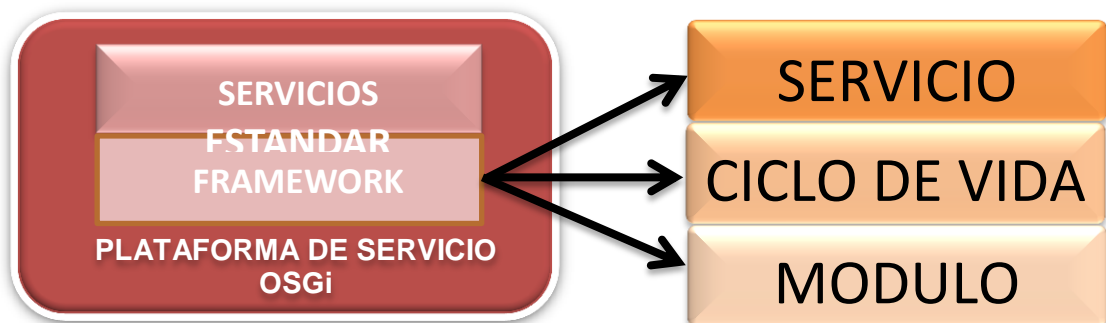
3.2. ARQUITECTURA SOFTWARE DE OSGi

La plataforma de servicios OSGi está compuesta por su Framework y por unos servicios estándar. El Framework es el entorno de ejecución que proporciona e implementa la funcionalidad de OSGi, en cuanto a los servicios estándar son APIs reutilizables para tareas básicas como Logging (Registro de Eventos), ajuste de preferencias de los servicios, entre otros.

3.3 FRAMEWORK OSGi

El Framework OSGi es el ambiente de ejecución de las aplicaciones modulares y define tres capas conceptuales: La capa de módulo, la capa de ciclo de vida y la capa de servicio.

Figura 1: Arquitectura en capas del Framework OSGi



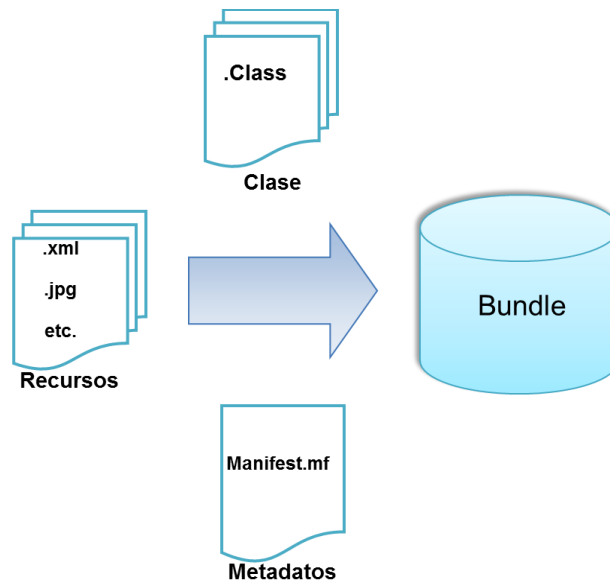
Fuente: OSGi IN ACTION CREATING MODULAR APPLICATIONS IN JAVA

3.3.1 CAPA MÓDULO

En esta capa se define el concepto de módulo OSGi, llamado también *Bundle*, donde un Bundle es un archivo JAR compuesto de:

- Recursos como clases, recursos web, entre otros; que implementen alguna funcionalidad o algún tipo de servicio (Si son requeridos para proveer algún servicio).
 - Un archivo MANIFEST.MF que describe el contenido del archivo JAR e información acerca del Bundle.
 - Una clase especial que debe implementar la interfaz BundleActivator (org.osgi.framework.BundleActivator), la cual permite que sean invocados los métodos para iniciar y detener el Bundle (start - stop).
-
- Puede tener distintos recursos opcionales, los cuales no son necesarios para la ejecución del Bundle, como documentación o código fuente. (Ver Figura 2).

Figura 2: *Bundle* con su respectivo código, recursos y metadatos



Fuente: OSGi IN ACTION CREATING MODULAR APPLICATIONS IN JAVA

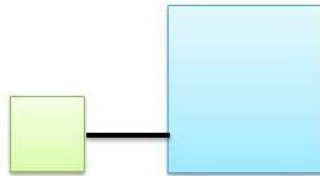
Los *Bundles* son una estructura más completa que los JAR convencionales porque pueden declarar explícitamente qué paquetes contenidos son

visibles externamente, complementando los modificadores de acceso básicos del lenguaje Java (público, privado y protegido). Adicionalmente es posible definir los paquetes de los que externamente depende cada *Bundle*, donde el principal beneficio es que el Framework OSGi administra y verifica la consistencia de las dependencias antes de iniciar los Bundles, garantizando que la aplicación no va a acceder a clases que actualmente no están disponibles, ya que la resolución de dependencias es realizada antes de la inicialización de los *Bundles* [9] [10].

A continuación se explica la visualización interna y externa del código para los Bundles.

EXPORTAR PAQUETE: Para exportar paquetes se declara en el archivo de manifiesto (manifest), con la cabecera `Export-Package` y el nombre de los paquetes que se desean exportar, estos quedan disponibles para que cualquier Bundle en la plataforma los acceda y los utilice.

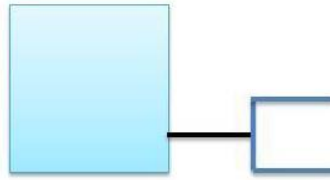
Figura 3: Descripción de un paquete exportado



Fuente: OSGi IN ACTION CREATING MODULAR APPLICATIONS IN JAVA

IMPORTAR PAQUETE: Para importar paquetes se declara en el manifest, con la cabecera `Import-Package` se define qué paquete específico se desea importar.

Figura 4: Descripción de la importación de un paquete [5]



Fuente: OSGi IN ACTION CREATING MODULAR APPLICATIONS IN JAVA

Para OSGi, los metadatos para importar código externo son importantes, porque es la forma como el Framework sabe lo que necesita cada Bundle.

3.3.2 CAPA DE CICLO DE VIDA

La capa de ciclo de vida define cómo se instalan y administran dinámicamente los *Bundles* en el Framework OSGi. Este ciclo de vida a nivel externo a la aplicación, define para cada *Bundle* las operaciones de instalación, actualización, inicio, parada y desinstalación; permitiendo administrar y evolucionar las aplicaciones, a su vez garantizando la adición y remoción de *Bundles* sin tener que reiniciar la aplicación. A nivel interno a la aplicación determina cómo accede cada uno de los *Bundles* al entorno de ejecución e interactúa con el Framework OSGi [5].

Como se puede apreciar en la figura 5, un *Bundle* puede encontrarse en cualquiera de los siguientes estados [9] [11]:

- Instalado: El Bundle se instaló en el Framework pero aún no se puede utilizar debido a la falta de dependencias.
- Resuelto: Todas las clases del *Bundle* incluidas las importadas están disponibles y está listo para ejecutarse.
- Iniciando: El *Bundle* ha sido iniciado y es llamado el método *start* de la clase que implementa la interfaz *BundleActivator*.
- Activo: El *Bundle* se ejecuta exitosamente.
- Detenido: El *Bundle* deja de ejecutarse y es llamado el método *stop* de la clase que implementa la interfaz *BundleActivator*.

- Desinstalado: El *Bundle* fue desinstalado del Framework.

Figura 5: Ciclo de vida de un Bundle



Fuente: OSGi IN ACTION CREATING MODULAR APPLICATIONS IN JAVA

3.3.3 CAPA DE SERVICIO

La capa de servicio soporta y proporciona un modelo de programación de aplicaciones flexible que incorpora conceptos como la computación orientada a servicios. La capa de servicio de OSGi proporciona un enfoque de desarrollo basado en interfaces, que separa la interfaz de su implementación, proporcionando a través de las interfaces Java un contrato entre los proveedores de servicios y los clientes de los servicios [5] [10].

Adicionalmente la capa de servicios expande a la capa de ciclo de vida, brindando dinamismo a los servicios y permitiéndoles aparecer o desaparecer en cualquier momento.

Un servicio es un objeto que proporciona funcionalidad. Es registrado por el *Bundle* en el Framework para ser usado por otros *Bundles*. Un *Bundle* puede reutilizar funcionalidad del sistema, importando servicios disponibles, pero debe ser controlado para evitar dependencia entre todos los *Bundles* del sistema, ocasionando pérdida de modularidad.

Figura 6: La interacción orientada a servicios. Proveedores de servicios publican en un registro donde los solicitantes pueden descubrir cuáles son los servicios disponibles para su uso.



Fuente: OSGi IN ACTION CREATING MODULAR APPLICATIONS IN JAVA

Un servicio se realiza mediante la creación de una interfaz, que debe definir los diferentes métodos a proveer como servicio. Para que un *Bundle* pueda acceder a los servicios disponibles en el Framework debe realizarlo mediante una instancia del *BundleContext*. El *BundleContext* representa el entorno de ejecución específico del *Bundle*, que es creado cuando el *Bundle* es instalado en el Framework [11] [12].

3.4 VAADIN

Vaadin es una biblioteca de Java que está diseñada para facilitar la creación y mantenimiento de interfaces web de usuario de alta calidad. La clave de Vaadin es el modelo de programación Server-Driven, que permite la creación de interfaces de programa de usuario al igual que cualquier aplicación de escritorio Java con herramientas convencionales, tales como AWT, Swing o SWT, pero de una manera más fácil.

Con el modelo Server-Driven, Vaadin se encarga de la gestión de la interfaz de usuario en el navegador y las comunicaciones *AJAX* entre el navegador y el servidor. Con el enfoque de Vaadin, no es necesario aprender y depurar tecnologías de navegación, tales como HTML o JavaScript.

3.3.1 ARQUITECTURA DE VAADIN

Vaadin proporciona dos modos de desarrollo para aplicaciones web: para el lado del cliente que se ejecuta en el navegador como un programa de JavaScript y para el lado del servidor. El modelo de desarrollo dirigido por el servidor es el más potente, permitiendo el desarrollo de la aplicación exclusivamente desde el lado del servidor, utilizando el motor del lado del cliente de Vaadin basado en AJAX que renderiza la interfaz de usuario en el navegador. El modelo del lado del cliente permite desarrollar controles y aplicaciones en Java, que son compilados a JavaScript y ejecutados en el navegador. Los dos modelos pueden compartir los controles UI, los temas, y el código de respaldo y servicios, y pueden ser mezclados juntos fácilmente.

Vaadin cuenta con el apoyo de GWT para una amplia gama de navegadores, por lo que el desarrollador no tiene que preocuparse acerca de la compatibilidad del

navegador. Detrás del modelo de servidor, Vaadin usa de la mejor manera las técnicas AJAX (*Asynchronous JavaScript and XML*) que hacen posible la creación de Aplicaciones de Internet enriquecidas (RIA) que son tan sensibles e interactivas como las aplicaciones de escritorio.

En su interior, Vaadin utiliza *GWT*, el *Google Web Toolkit*, para generar la interfaz de usuario en el navegador. Los programas GWT están escritos en JAVA, pero compilados en JavaScript, liberando al programador de aprender JavaScript y otras tecnologías de navegador. GWT es ideal para la aplicación de avanzados componentes de la interfaz de usuario (o widgets en la terminología GWT) y la lógica de interacción en el navegador, mientras que Vaadin se encarga de la lógica de la aplicación real en el servidor. Vaadin está diseñado para ser extensible, y permite utilizar cualquiera de los componentes GWT con facilidad, además del repertorio de los componentes ofrecidos en Vaadin [13].

4. DESARROLLO DE LA PRÁCTICA EMPRESARIAL

4.1 APLICACIONES EMPRESARIALES EN BIOSYS LTDA.

La empresa Biosys Ltda. para el desarrollo de aplicaciones empresariales orientadas a la WEB integra los Frameworks Vaadin e Hibernate sobre la plataforma Java Enterprise Edition JEE.

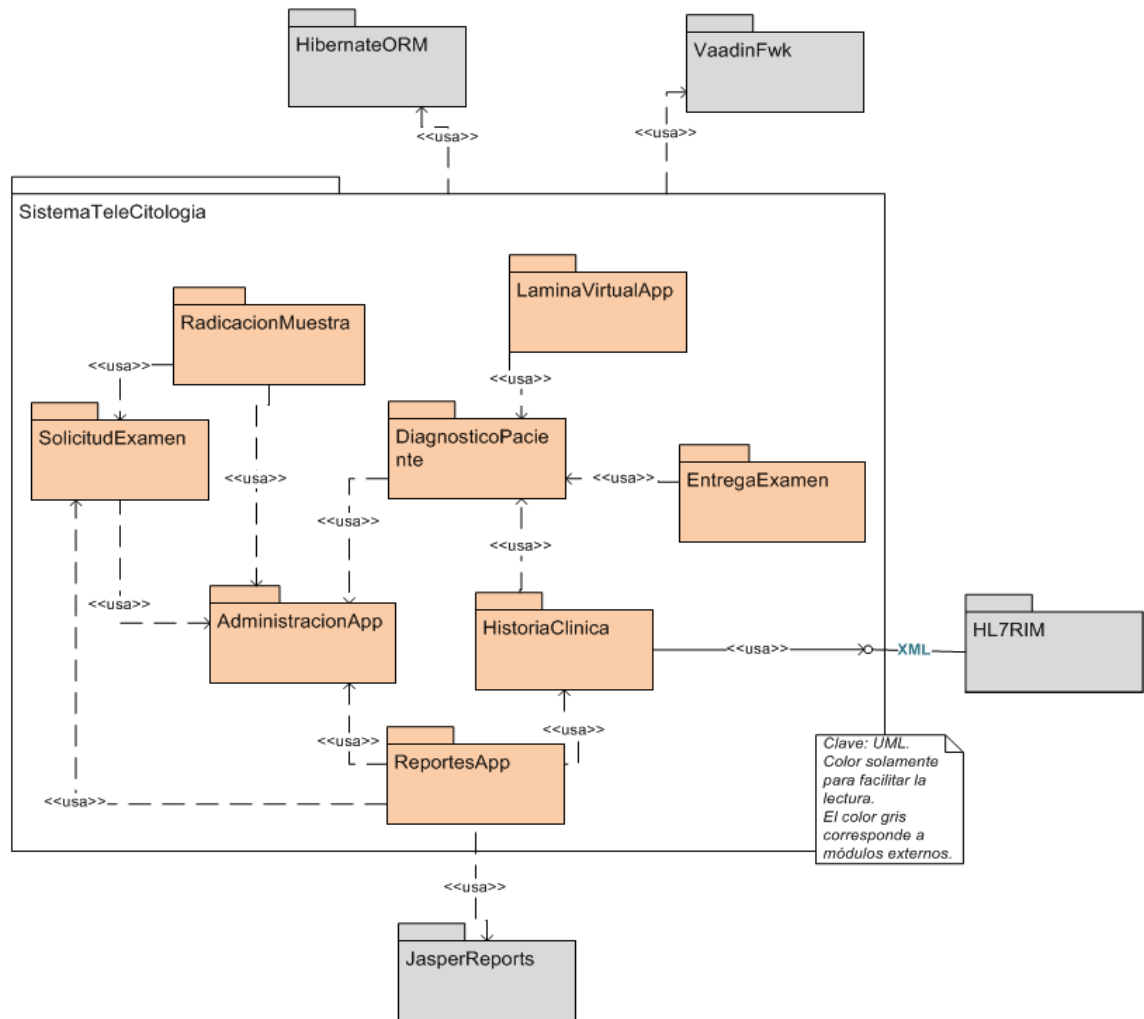
El Framework Vaadin permite el desarrollo de interfaces de usuario enriquecidas por medio de HTML5; incrementando la productividad en el desarrollo de aplicaciones ya que abstrae para el desarrollador las comunicaciones asíncronas AJAX, la operabilidad en diferentes navegadores web y dispositivos móviles, y aumenta la seguridad de las aplicaciones por medio de su arquitectura orientada a servidor manteniendo tanto la lógica del negocio como la lógica de la interfaz siempre del lado del servidor.

El Framework Hibernate por su parte, se encarga de administrar el acceso a los datos del modelo de negocio por medio de la especificación Java Persistence API (JPA) que realiza el mapeamento objeto relacional, integrando el modelo relacional y el paradigma orientado a objetos.

4.2 CITOSOFT: SISTEMA DE TELECITOLOGIA PARA APOYO AL PROGRAMA DE DETECCIÓN TEMPRANA DE CÁNCER CÉRVICOUTERINO EN COLOMBIA

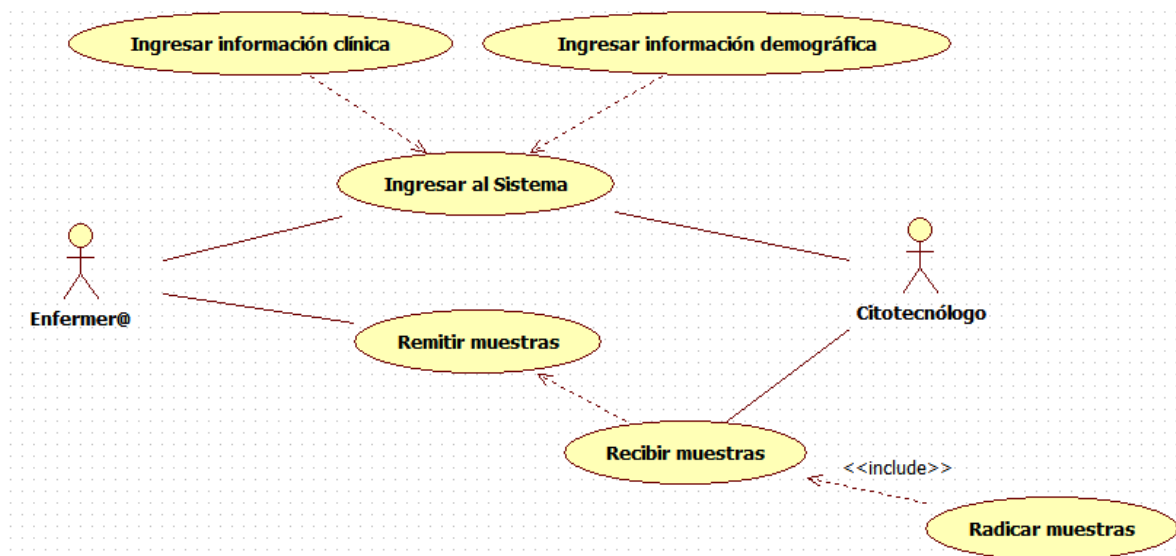
Soportado en estos dos Frameworks en Biosys Ltda. fue desarrollado el Sistema de Telecitología CITOSOFT que implementa la arquitectura propuesta por Lina María Garcés Rodríguez en su tesis de maestría titulada “Análisis y Diseño de un Sistema de Telecitología que apoye el programa de detección temprana de cáncer cérvicouterino en Colombia” [3].

Figura 7: VISTA DE USOS DE ALTO NIVEL DE CITOSOFT



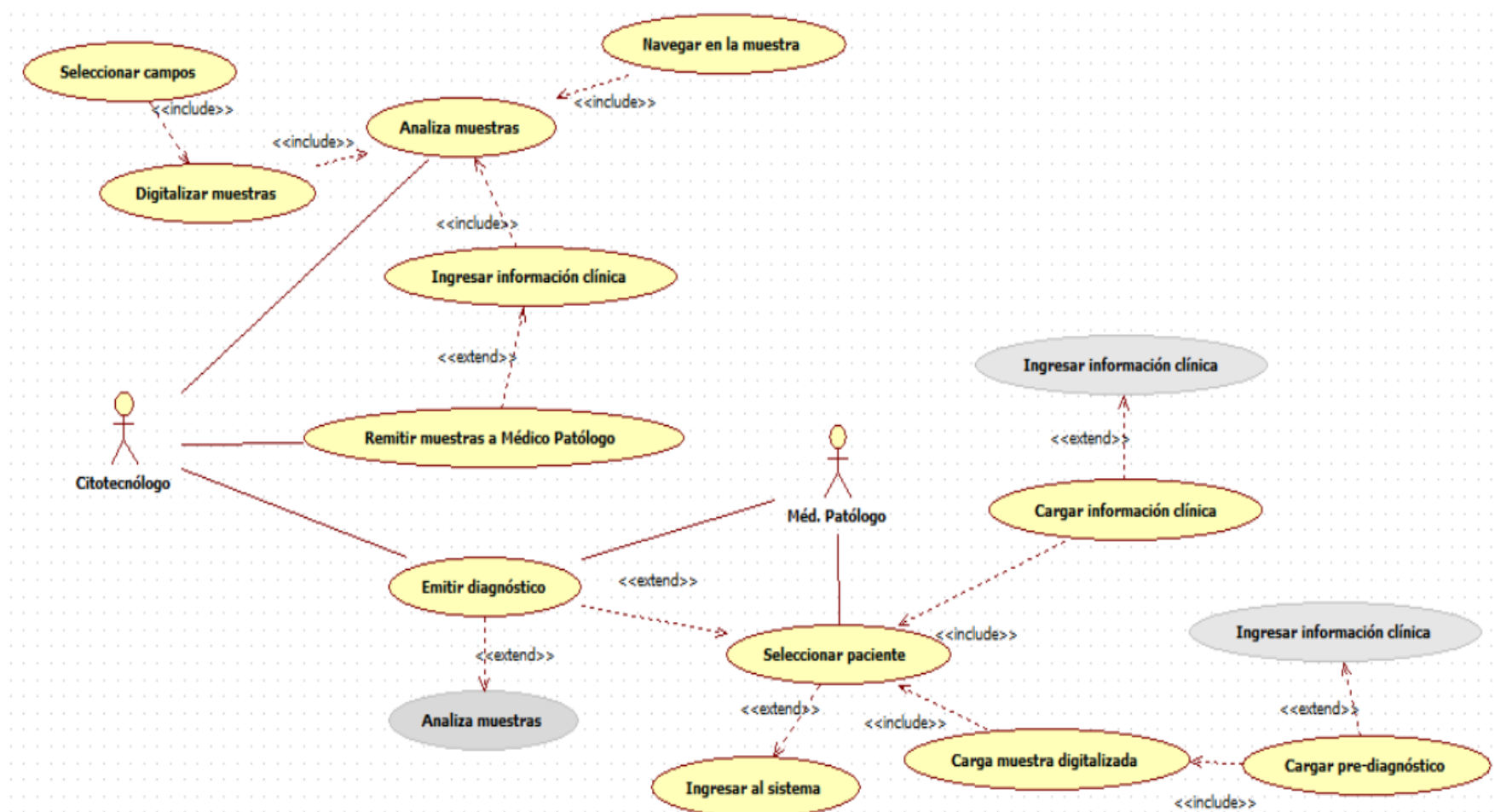
Fuente: ANÁLISIS Y DISEÑO DE UN SISTEMA DE TELECITOLOGIA QUE APOYE EL PROGRAMA DE DETECCIÓN TEMPRANA DE CANCER CERVICOUTERINO EN COLOMBIA [RECURSO ELECTRÓNICO] / LINA MARIA GARCES R.; DIRECTORES LOLA XIOMARA BAUTISTA ROZO, OLGA M., ALVAREZ O.

Figura 8: Casos de Uso para la Fase de Pre Análisis del Proceso de Diagnóstico basado en Citología Cérvicouterina.



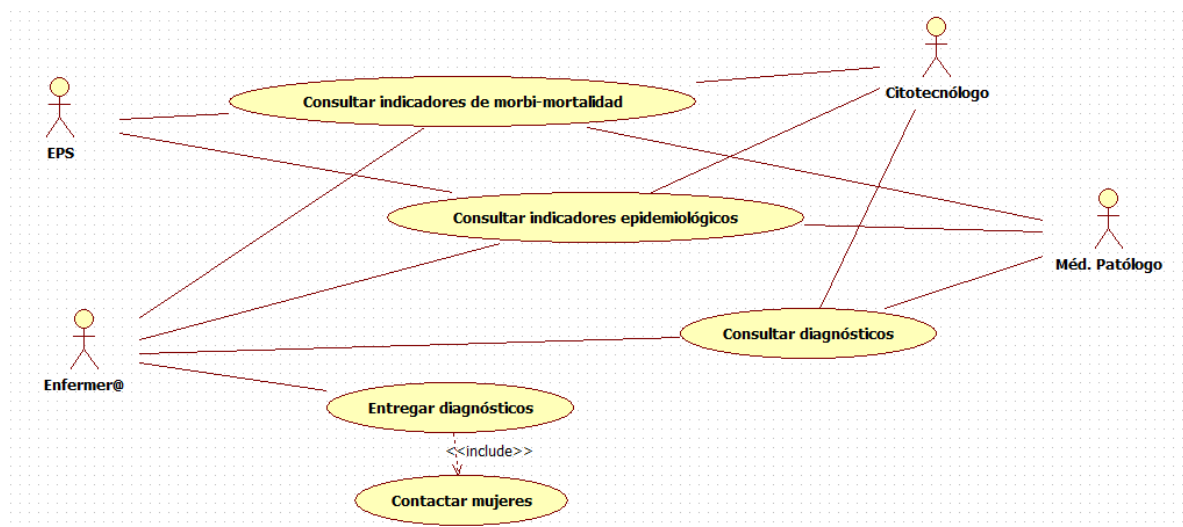
Fuente: ANÁLISIS Y DISEÑO DE UN SISTEMA DE TELECITOLOGIA QUE APOYE EL PROGRAMA DE DETECCIÓN TEMPRANA DE CANCER CERVICOUTERINO EN COLOMBIA [RECURSO ELECTRÓNICO] / LINA MARIA GARCES R.; DIRECTORES LOLA XIOMARA BAUTISTA ROZO, OLGA M., ALVAREZ O.

Figura 9: Casos de Uso para la Fase de Análisis del Proceso de Diagnóstico basado en Citología Cérvicouterina



Fuente: ANÁLISIS Y DISEÑO DE UN SISTEMA DE TELECITOLOGIA QUE APOYE EL PROGRAMA DE DETECCIÓN TEMPRANA DE CANCER CERVICOUTERINO EN COLOMBIA [RECURSO ELECTRÓNICO] / LINA MARIA GARCES R.; DIRECTORES LOLA XIOMARA BAUTISTA ROZO, OLGA M., ALVAREZ O.

Figura 10: Casos de Uso para la Fase de Pós Análisis del Proceso de Diagnóstico basado en Citología Cérvicouterina .



Fuente: ANÁLISIS Y DISEÑO DE UN SISTEMA DE TELECITOLOGIA QUE APOYE EL PROGRAMA DE DETECCIÓN TEMPRANA DE CANCER CERVICOUTERINO EN COLOMBIA [RECURSO ELECTRÓNICO] / LINA MARIA GARCES R.; DIRECTORES LOLA XIOMARA BAUTISTA ROZO, OLGA M., ALVAREZ O.

A partir de los requisitos de la arquitectura, la vista de usos de alto nivel de la figura 7 y los casos de uso resumidos en las figuras 8,9,10 fueron desarrollados los módulos:

- **Administración de Módulos:**

Permite el registro de los diferentes módulos del sistema estableciendo una relación entre los casos de uso de la aplicación y los niveles de acceso de los usuarios al sistema. A partir del registro de los módulos es realizado el gerenciamiento, personalización y navegación de los usuarios en el sistema de acuerdo a los roles habitados en las diferentes unidades de negocio.

- **Administración de Usuarios:**

Módulo para el registro de usuarios en el sistema. Cuando es accedido por un usuario con rol administrador es posible cambiar información personal de los diferentes usuarios así como habilitar o inhabilitar su acceso al sistema. Cuando es accedido por un usuario que aún no se ha registrado en el sistema, le permite registrarse en el sistema y este solo tendrá acceso luego de validación de su información personal de acuerdo al rol a ejercer en el sistema.

- **Administración de Procesos:**

Permite agrupar servicios en unidades mayores denominadas Procesos. Los procesos son grupos de actividades (Módulos) desempeñadas por los diferentes roles en el sistema. Los módulos que se agrupan usualmente están relacionados en la lógica del negocio de la aplicación.

Ejemplo:

Proceso de Toma muestras

- Módulo de Registro Clínico de Pacientes
- Módulo de consulta de diagnósticos.

- **Administración de Roles:**

Los roles permiten agrupar a los usuarios de acuerdo a las actividades que ejecutan en el sistema. Un usuario puede pertenecer a más de un rol en cada una de las unidades de negocio. En este módulo se definen los procesos que puede realizar cada uno de los roles del sistema.

- **Administración de Unidades de Negocio:**

Las Unidades de Negocios son todas las instituciones que consumen servicios de CITOSOFT. En este módulo son habilitadas a las unidades de negocio paquetes de módulos denominados procesos de acuerdo a los servicios previamente contratados en CITOSOFT. Cuando este módulo del sistema es accedido por un usuario responsable de administrar una Unidad

de Negocio, es posible personalizar los usuarios que desempeñarán roles específicos dentro de la unidad de negocio seleccionada.

- **Navegación:**

Módulo responsable de personalizar el menú del usuario dependiendo de las unidades de negocio que han contratado sus servicios así como los roles que ejerce en cada una de estas.

- **Inicio de Sesión:**

Módulo que valida la información del usuario y desencadena la ejecución del módulo de navegación sólo cuando los datos de acceso son correctos.

- **Registro Clínico de Pacientes:**

Módulo que permite registrar los datos personales del paciente, lesiones previas, datos clínicos relevantes relacionados a los exámenes citológicos e información referente a su afiliación al Sistema de Seguridad Social.

- **Radicación de Muestras:**

Módulo que permite a los citotecnólogos radicar y agregar al inventario de la unidad de negocio muestras citológicas para su correspondiente digitalización y análisis.

- **Digitalización de Muestras citológicas:**

Permite la digitalización de muestras citológicas por medio de una aplicación embebida en el navegador, que emplea el protocolo JNLP para garantizar la ejecución de la versión más reciente en el cliente. Esta aplicación embebida digitaliza las regiones visualizadas por el citotecnólogo, alinea las imágenes y las transmite al servidor para un posterior análisis por médicos patólogos.

- **Registro y diagnóstico de lesiones citológicas:**

Módulo para registro y emisión de diagnósticos citológicos por parte del citotecnólogo y el médico patólogo de las lesiones citológicas de las muestras analizadas, que son clasificadas de acuerdo al sistema Bethesda [3].

- **Consulta de diagnósticos:**

Permite la visualización de los diagnósticos emitidos por los especialistas.

- **Indicadores de morbi-mortalidad y epidemiológicos:**

Genera reportes de las lesiones encontradas en los pacientes discriminados por la Institución Prestadora de Servicios (IPS), Empresa Prestadora de Servicios (EPS), Departamento o Municipio.

4.3. LIMITACIONES DE LA IMPLEMENTACIÓN DE CITOSOFT

El sistema CITOSOFT a pesar de tener claramente definidos los módulos como fue expuesto en la sección anterior y ser implementados en paquetes (packages) independientes, su despliegue en los servidores de aplicación como Tomcat, GlassFish o JBoss, es realizado por medio de un único archivo de aplicación conocido como WAR (**W**eb application **A**Rchive).

Este archivo WAR contiene toda la aplicación como una unidad y si es requerido algún cambio en alguno de los módulos desarrollados o insertar un nuevo módulo al sistema, es necesario empaquetar todos los cambios en un nuevo archivo WAR y efectuar la correspondiente actualización en el servidor de aplicaciones, que conlleva a interrupciones en el servicio. Esta es una de las razones, por la cual, las actualizaciones de la mayoría de las aplicaciones empresariales son realizadas en periodos de tiempo de baja demanda de usuarios como fines de semana u horas

de la madrugada; y si un cambio que desea realizarse es considerado crítico, es inevitable una interrupción completa del funcionamiento de la aplicación.

Adicionalmente al no existir un control de versiones independiente para cada módulo del sistema con una especificación exacta de sus dependencias; cambios realizados en uno de los módulos de la aplicación podrían afectar la funcionalidad de otros módulos fuertemente acoplados y por tanto generar en tiempo de ejecución comportamientos inesperados de la aplicación.

A pesar de la versión empresarial de Java JEE de intentar abordar el problema de modularización por medio de empaquetamiento tipo EAR (Enterprise ARchive) donde es posible encapsular módulos de la aplicación y crear aplicaciones como una composición de módulos en que se reutilizan módulos previamente desarrollados[14]; esta modularidad sólo es útil en tiempo de desarrollo debido a que luego de desplegada la aplicación, cualquier cambio realizado implica detener la aplicación que se encuentra actualmente en ejecución y ser reemplazada por la versión mejorada.

Actualmente con el objetivo de mejorar la modularidad de las aplicaciones JSE y JEE así como para modularizar la plataforma JAVA fue creado el proyecto Jigsaw, amparado en los documentos de mejora al JDK (Java Development Kit) JEP 200[15] y JEP 201[16]. Ese proyecto fue propuesto para ser incorporado desde la versión 7 de la plataforma JAVA en 2011, siendo retrasado para la versión 8 y posteriormente para la versión 9 que será desplegada para el año 2015. Este proyecto busca abordar el mismo problema de modularidad de OSGi, por tal razón la misma comunidad que brinda soporte a la plataforma Java creó un proyecto complementario llamado Penrose para explorar e implementar la interoperabilidad entre OSGi y Jigsaw.

Como aún el futuro del proyecto Jigsaw es incierto y una amplia aceptación por parte de la comunidad podría tardar unos años más si es desplegado en el año 2015, fue seleccionado OSGi en Biosys Ltda. como arquitectura modular para el desarrollo de aplicaciones empresariales, apostando por la interoperabilidad prometida por el proyecto Penrose hacia la plataforma Java, que estaría modularizada bajo la arquitectura propuesta por el proyecto Jigsaw.

4.4 MODULARIZACIÓN DE CITOSOFT EN OSGI

En el proceso de modularización del sistema CITOSOFT se consideraron los siguientes factores que permitieron definir las diferentes fases para el cumplimiento de los objetivos propuestos en la práctica empresarial:

- Cuál implementación de la arquitectura OSGi que brinde soporte a tecnologías de la plataforma JEE debe ser utilizada?
- Como extender el Framework Vaadin y dar soporte a la capa de servicios de OSGi?
- Como estructurar los diferentes paquetes (packages) de los módulos de Biosys Ltda. y de esta manera transformar y desacoplar a CITOSOFT en unidades independientes conocidas como Bundles?

Estos factores serán abordados en las próximas secciones, profundizando en las implementaciones realizadas en la práctica empresarial.

4.5. ARQUITECTURA OSGI EN BIOSYS LTDA.

Uno de los objetivos específicos de la práctica empresarial era definir qué niveles de la arquitectura OSGi deberían ser implementados para dar soporte dinámico a las aplicaciones en Biosys Ltda.

Luego de una revisión a la versión 5 de la especificación OSGi se determinó que era requerida una implementación completa de los niveles de la arquitectura OSGi

debido a que el dinamismo requerido para el despliegue de CITOSOFT sólo se obtendría en el nivel de servicios de la arquitectura, que conlleva a la implementación de todos los niveles subyacentes. Una implementación de esta naturaleza está fuera del alcance de una práctica empresarial de pregrado y se concluyó que cualquiera de las implementaciones existentes podría satisfacer las necesidades requeridas. Dentro de las opciones disponibles las de mayor aceptación en la comunidad son Apache Felix y Eclipse Equinox.

Como el desarrollo de aplicaciones empresariales requiere un conjunto de características adicionales que extienden la especificación original de OSGi y que en su mayoría son proporcionadas por la plataforma JEE; en la versión 5 de OSGi fue adicionada una especificación complementaria llamada Especificación Empresarial OSGi Versión 5, que adiciona soporte a JDBC (Java Database Connectivity), JPA (Java Persistence Api), JTA (Java Transaction Api), JMX (Java Management Extension), Blueprint, entre otros. A partir de la especificación empresarial fueron desarrolladas por la comunidad las implementaciones Apache Aries y Eclipse Gemini que complementan respectivamente a Apache Felix y Eclipse Equinox.

A partir de estas implementaciones se han desarrollado servidores OSGi como Apache Karaf², Apache Geronimo³, WebSphere⁴, Eclipse Virgo⁵, GlassFish⁶, JBoss, entre otros.

Por recomendación de Holly Cummins y Timothy Ward autores del libro Enterprise OSGi in Action [17], expertos con amplia trayectoria en el desarrollo de aplicaciones OSGi, pertenecientes a los grupos OSGi Alliance Core Platform y

² <http://karaf.apache.org/>

³ <http://geronimo.apache.org/>

⁴ <http://www-03.ibm.com/software/products/en/appserv-was>

⁵ <http://www.eclipse.org/virgo/>

⁶ <https://glassfish.java.net/>

Enterprise Expert, responsables de la especificación OSGi; que indican que es una buena práctica seleccionar un servidor de aplicaciones que proporcione preconfiguradas e integradas todas las tecnologías empresariales así como el núcleo de OSGi. En Biosys Ltda. fueron considerados únicamente los servidores Apache Geronimo y Eclipse Virgo que disponen de una implementación casi completa de la especificación empresarial de Java bajo licencias open source, amigables con start-ups.

Tanto Apache Gerónimo como Eclipse Virgo brindan las tecnologías necesarias para el desarrollo de aplicaciones empresariales en Biosys Ltda.; por este motivo fue seleccionado Apache Geronimo procurando en lo posible no emplear tecnologías fuera de las especificaciones de OSGi, lo que permitirá cambiar para Eclipse Virgo u otro Servidor de aplicaciones con licencia propietaria que adicione mejoras en rendimiento o consumo de recursos en el momento que se necesite optimizar el desempeño de la aplicación.

4.6 VAADIN Y OSGI

Actualmente el Framework Vaadin para desarrollo de aplicaciones Web ofrece soporte a la capa módulo de la arquitectura OSGi y está dividido en bundles que poseen manejo de versiones semántico.

Este soporte a OSGi no aprovecha el dinamismo ofrecido por la arquitectura y por tal razón las aplicaciones Web realizadas con Vaadin no cumplen con los lineamientos requeridos para el despliegue dinámicos de aplicaciones.

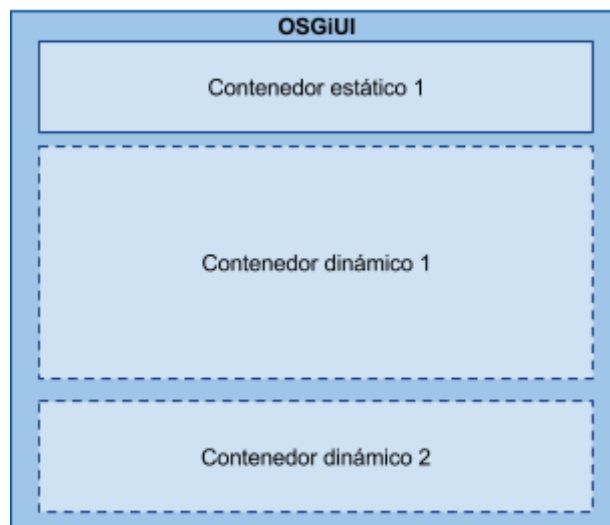
Durante el desarrollo de la práctica empresarial fue propuesta una extensión al framework de Vaadin que permite la construcción dinámica de interfaces usuario mediante utilización del sistema de registro de servicios de OSGi.

4.6.1 EXTENSIÓN DE VAADIN

Con el objetivo de permitir a las aplicaciones realizadas con el Framework Vaadin la construcción interfaces por agregación dinámica de servicios de OSGi fue extendida la clase [UI](#) de Vaadin, responsable de la gestión de los componentes en las interfaces de usuario.

Esta extensión de la clase UI del Framework Vaadin fue llamada en el api OSGiUI y adiciona soporte al nivel de servicios de la arquitectura OSGi e incorpora un modelo simple de diseño de interfaz para contenedores dinámicos que alojan los servicios de OSGi por medio de anotaciones.

Figura 11: Ejemplo de Interfaz dinámica en Vaadin con soporte a contenedores dinámicos.



Un ejemplo en Vaadin con soporte a OSGi con dos contenedores dinámicos como el de la figura 11 sería:

```
1 @UI(defaultUIServiceContainer="contenedorDinamico1")
2 public class Ejemplo1 extends OSGiUI{
3
4     VerticalLayout contenedorEstatico;
5
6     @UIServiceContainer(defaultUIService = "servicio1",name = "contenedorDinamico1")
```

```

8   VerticalLayout contenedorDinamico1;
9
10  @UIServiceContainer(defaultUIService = "servicio2",name = "contenedorDinamico2")
12  VerticalLayout contenedorDinamico2;
13  ...
14}

```

En este ejemplo en la línea 1 es adicionada la anotación `@UI` que permite definir a cuál de los contenedores dinámicos de la interfaz de usuario será direccionada una solicitud actualización de contenido dinámico cuando no se indique un contenedor específico.

En las líneas 6 y 10 es agregada la anotación `@UIServiceContainer` para indicar que los contenedores de tipo `VerticalLayout` van a renderizar servicios dinámicos de OSGi. Adicionalmente es indicado el nombre del contenedor y el servicio que será renderizado por defecto el cual será utilizado para inicializar la interfaz de usuario.

Para gestionar los diferentes servicios de la interfaz dinámica y garantizar un comportamiento independiente por usuario, OSGiUI instancia un administrador de interfaz `UIServiceManager`, desarrollado en la práctica empresarial, que gestiona cuáles interfaces de usuario deben ser notificadas y posteriormente actualizadas sin afectar a usuarios que no requieren de las actualizaciones.

Como en el registro de servicios de OSGi pueden existir toda clase de servicios, para que el administrador de interfaz gestione únicamente los servicios relacionados a la interfaz de usuario fue desarrollada la interfaz `UIService`.

```

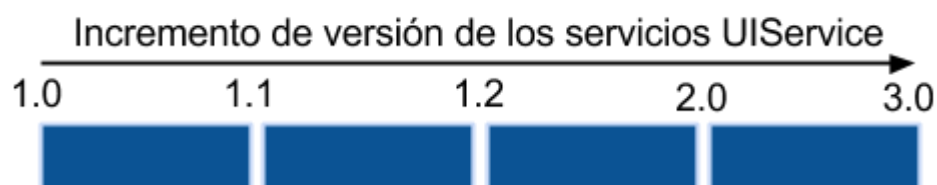
public interface UIService {
    public String getName();
    public Version getVersion();
    Class<? extends View> getView();
}

```

La interfaz *UIService* permite registrar extensiones de interfaz de usuario en el registro de servicios OSGi que posteriormente serán gestionadas por OSGiUI. Cada extensión posee un *nombre* que la identifica en el administrador de interfaz, una *versión* que permite determinar cuándo un servicio debe ser actualizado de forma tardía o explícita y una *vista* que será instanciada y renderizada en uno de los contenedores dinámicos.

La versión del servicio es independiente de la versión del Bundle. Esta versión a diferencia de la versión semántica del Bundle posee dos componentes numéricos. El primer componente llamado número mayor (*majorNumber*) cuando es incrementado indica que este servicio reemplazará de forma explícita a un servicio que posea su mismo nombre; y el segundo componente llamado número menor (*minorNumber*) cuando es incrementado indica que este servicio solo será visualizado en su versión más reciente por usuarios que al momento de la actualización no estén trabajando con el servicio; y por consiguiente los usuarios que actualmente desarrollan operaciones basadas en el servicio serán actualizados a la versión más reciente cuando terminen la transacción que están ejecutando, lo cual hemos denominado actualización tardía.

Figura 12: Ejemplo de versionamiento de servicios UIService



En la figura 12 es presentado un ejemplo de versionamiento para un servicio. Si el servicio en primera instancia es desplegado en el entorno OSGi con versión 1.0, al no existir una versión previa del servicio, esta versión es indicada como la versión de referencia y desplegada en las interfaces de usuario en las cuales los

contenedores indiquen que el servicio debe ser renderizado por defecto. Si una nueva versión del servicio (Versión 1.1) es adicionada al registro de servicios de OSGi y esta es mayor que la versión de referencia (Versión 1.0) se analizan sus componentes para determinar si la nueva versión será actualizada de forma explícita o tardía. En este caso será una actualización tardía pues únicamente fue incrementado el número menor de versión. Para que se desencadene una actualización explícita debe existir una diferencia mayor que cero en la componente de número mayor entre la versión a ser considerada y la versión de referencia. Un ejemplo de actualización explícita sería entre la versión (2.0) y la versión de referencia (1.2), en donde todos los usuarios que estén realizando operaciones con una versión inferior a la (2.0) serán notificados y posteriormente actualizados a la versión (2.0), cancelando sus operaciones en ejecución y ubicados en el estado inicial del caso de uso relacionado al servicio.

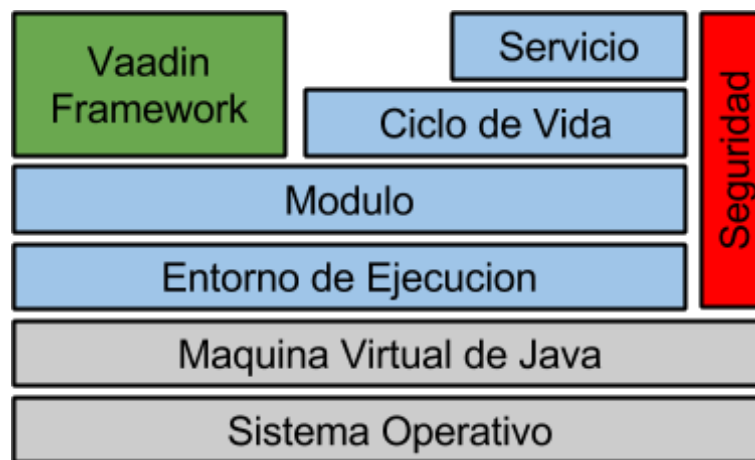
Otras extensiones han sido propuestas por parte de la comunidad de Vaadin que dan soporte al sistema de registro de servicios de OSGi pero ninguna de estas implementaciones aborda el problema de gestión multiusuario y actualización tardía. Estas implementaciones sólo se remiten a sincronizar la interfaz usuario basadas en los servicios disponibles en el servicio de registro sin proponer estrategias de interrupción y/o reemplazo de los servicios que actualmente son ejecutados por los usuarios y de esta manera mitigar el impacto de la actualización. Adicionalmente algunas de estas implementaciones presentan problemas en entornos multiusuario, donde los componentes desplegados para diferentes usuarios son compartidos, ocasionando problemas de ejecución en las aplicaciones Vaadin debido a que los componentes sólo pueden pertenecer a un único contenedor.

Otra contribución importante al Framework de Vaadin fue el soporte a supervisores de solicitudes y respuestas del Framework. Los supervisores de solicitudes en la especificación de la plataforma JAVA EE son implementados a

través de *filtros* que interceptan las solicitudes HTTP antes de ser procesadas y permiten inicializar configuraciones como conexiones a bases de datos o mejorar la seguridad de las aplicaciones. En Vaadin es posible crear y registrar filtros por medio de la plataforma JEE pero estos no interceptan solicitudes realizadas por WebSockets, proporcionando un soporte incompleto a los supervisores. Para resolver ese problema del Framework Vaadin, en la práctica empresarial se realizó una extensión de la clase *VaadinServlet* en la que se unifica la interceptación de solicitudes, permitiendo a los supervisores de solicitudes en Biosys gestionar de manera eficiente las conexiones a la base de datos.

Adicionalmente fue desarrollado en la práctica empresarial un módulo con nombre `com.biosys.bundlemanager` que empleando la extensión propuesta del Framework de Vaadin controla el ciclo de vida de los todos los *Bundles* de la plataforma y permite mediante una interfaz gráfica instalar, remover, detener e iniciar *Bundles*.

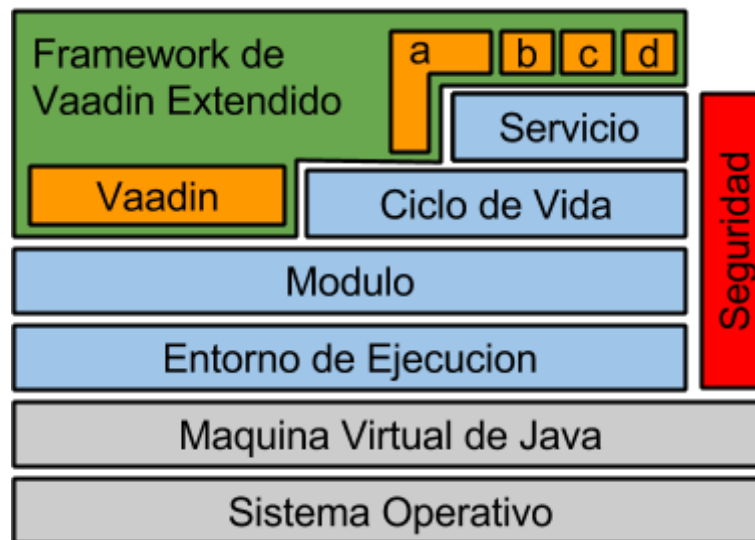
Figura 13: Soporte por defecto del Framework Vaadin a la arquitectura OSGi



En la figura 13 es mostrado el soporte por defecto del Framework Vaadin al modelo de capas de la arquitectura OSGi, en que los Bundles que componen el framework Vaadin están asociados únicamente al nivel Módulo de la arquitectura.

En la figura 14 es presentado el Framework de Vaadin extendido resultante de la práctica empresarial. La letras (a), (b), (c) y (d) hacen referencia a los Bundles desarrollados `com.biosys.bundlemanager`, `com.biosys.uicontroller`, `com.biosys.vaadinapplicationmanager` y `com.biosys.vaadininterceptor` respectivamente.

Figura 14: Soporte del Framework de Vaadin Extendido a la arquitectura OSGi



4.7 MÓDULOS DE CITOSOFT

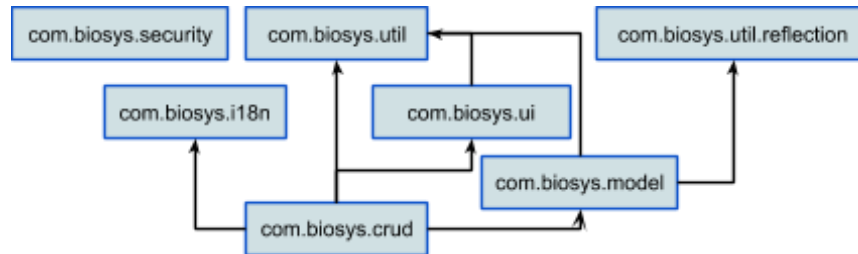
NÚCLEO DE LAS APLICACIONES EMPRESARIALES EN BIOSYS LTDA.

Con lo enunciado en las secciones anteriores cabe notar que el objetivo de este proyecto es prever posibles dificultades en su fase de implantación fue adaptado en la práctica empresarial el sistema CITOSOFT a una arquitectura modular soportada en el Framework OSGi, que integra las características del Framework Vaadin y simplifica la fase de mantenimiento de software, permitiendo actualizar componentes individuales sin afectar el funcionamiento global del sistema. Es por esto que en la de modularización de CITOSOFT fueron analizadas las dependencias entre los diferentes paquetes que conforman los módulos del

sistema. Se observó que existía un conjunto de paquetes denominado el núcleo de las aplicaciones empresariales en Biosys Ltda., encargados de implementar las operaciones esenciales de acceso al modelo de datos, diseño de interfaz gráfica, formato de datos, entre otros. Estos paquetes son detallados a continuación:

- **com.biosys.ui:**
Conjunto de componentes de interfaz de usuario que extienden la funcionalidad ofrecida por Vaadin.
- **com.biosys.security:**
Paquete encargado del cifrado de mensajes.
- **com.biosys.i18n:**
Paquete responsable de la internacionalización de las aplicaciones
- **com.biosys.util:**
Paquete utilizado para la manipular cadenas, conversión de tipos de datos y operaciones con fechas.
- **com.biosys.util.reflection:**
Paquete que permite la manipulación de anotaciones y acceso a atributos y métodos de las clases indirectamente por medio de la Api de Reflection de Java
- **com.biosys.model:**
Paquete que extiende al Framework Hibernate y brinda soporte en las operaciones de acceso a datos.
- **com.biosys.crud:**
Paquete que unifica la interfaz de usuario para las operaciones de crear, leer, actualizar y eliminar conocidas por el acrónimo CRUD (**C**reate, **R**ead, **U**ppdate and **D**elete), facilitando su extensión para nuevas clases adicionadas a modelo de datos.

Figura 15: Diagrama de dependencias de los paquetes del núcleo de las aplicaciones Biosys.



Como primera etapa en la modularización los paquetes del núcleo de Biosys fueron convertidos a *Bundles*, razón por la cual en esta sección del documento los conceptos *Bundle* y paquete son tratados indistintamente.

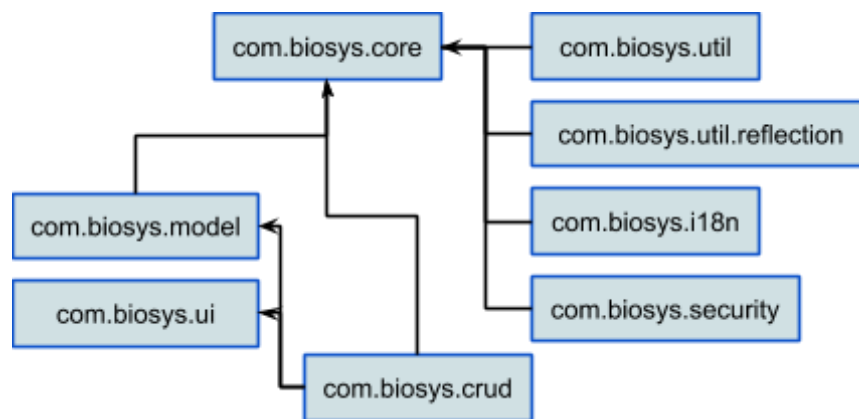
Dar soporte al nivel módulo de la arquitectura OSGi no implica un gran esfuerzo a nivel de desarrollo de software, sin embargo debe ser realizado con bastante cuidado debido a que los paquetes considerados públicos serán accedidos por los demás *Bundles* del sistema; y si es realizada una publicación deliberada puede ocasionar que el sistema resultante sea fuertemente acoplado.

Para evitar un acoplamiento fuerte se deben realizar implementaciones en los *Bundles* del sistema que adicionen funcionalidad en niveles superiores de la arquitectura de OSGi como el nivel de ciclo de vida y el nivel de servicios.

La figura 15 muestra la dependencia de los paquetes del núcleo de biosys antes de la modularización. En esta implementación todas las clases de los diferentes paquetes se encuentran expuestas y si es requerida una actualización en alguno de los paquetes, el reemplazo a la nueva versión desencadenará que todos los paquetes dependientes sean reiniciados. Para mitigar el acoplamiento fue propuesto el *Bundle* **com.biosys.core** que define un api para la estandarización de los servicios de seguridad (`com.biosys.security`), utilidades (`com.biosys.util`),

reflexión (`com.biosys`) e internacionalización (`com.biosys.i18n`). Esta API propone diferentes interfaces que son accedidas por los *Bundles* que consumen los servicios (`com.biosys.crud`) así como los *Bundles* que los proporcionan (`com.biosys.util`, `com.biosys.i18n`, entre otros), creando un acoplamiento débil en tiempo de ejecución que es controlado por el sistema de registro de servicios de OSGi.

Figura 16: Dependencia de los paquetes del núcleo de Biosys después de la modularización



La figura 16 muestra la dependencia de paquetes del núcleo de OSGi luego de la modularización. En esta nueva versión los cambios en las implementaciones de los *Bundles* que proveen los servicios del API de `com.biosys.core` no afectan el funcionamiento de los bundles consumidores.

Para los *Bundles* `com.biosys.model` y `com.biosys.ui` no es posible realizar implementaciones en niveles superiores de la arquitectura de OSGi ya que las clases que exponen por medio de sus paquetes son extendidas o instanciadas directamente por otros Bundles del sistema.

4.8. ADMINISTRACIÓN DE APLICACIONES EN BIOSYS LTDA

Complementario a los módulos asociados al núcleo de las aplicaciones empresariales, Biosys Ltda. dispone de un conjunto de módulos denominado Módulos de administración de Negocios. Este conjunto de módulos es responsable de la administración de usuarios, módulos, procesos, roles, unidades de negocio, navegación e inicio de sesión de las aplicaciones desarrolladas en Biosys Ltda.

En la práctica empresarial basada en la extensión del Framework de Vaadin se modificaron los módulos de administración de aplicaciones, brindando soporte a las capas Módulo y Servicios de la arquitectura de OSGi. A partir de los cambios realizados los módulos pasaron a ser servicios de la plataforma OSGi siendo detectados por la plataforma en tiempo de ejecución e integrados al Sistema CITOSOFT.

4.9. MÓDULOS ASOCIADOS AL PROCESO DE DIAGNÓSTICO BASADO EN CITOLOGÍA CÉRVICOUTERINA

Al igual que los módulos de administración de aplicaciones fueron realizadas las correspondientes mejoras a los módulos de Diagnóstico basados en Citología Cérvicouterina, siendo convertidos en *Bundles* y publicados como servicios en el sistema de registro de servicios de OSGi. Los módulos que forman parte del proceso de Diagnóstico son la esencia del sistema CITOSOFT y fueron descritos por los casos de uso de las figura 8, 9 y 10.

4.10 PRUEBAS DE FUNCIONALIDAD DEL SISTEMA CITOSOFT

Como el objetivo de la práctica empresarial era la adaptación del sistema CITOSOFT a una arquitectura modular, son presentadas las pruebas funcionales y no funcionales relacionadas a los casos de uso y a los módulos desarrollados para

dar soporte al Framework OSGi. Posteriormente serán consideradas las pruebas de funcionalidad asociadas a los requerimientos de CITOSOFT presentados en la sección 4.2.

La adaptación del sistema a la arquitectura modular del Framework OSGi basado en la clasificación de [21] y [22] contribuyó al mejoramiento de los siguientes requerimientos no funcionales:

- **Fiabilidad:** En los atributos de calidad referentes a exactitud, tolerancia a fallos y recuperabilidad. En exactitud por medio del versionamiento semántico garantizando que los diferentes módulos del sistema solo podrán interactuar con versiones previamente definidas; tolerancia a fallos debido a que un módulo solo será desplegado cuando todas sus dependencias estén resueltas; y recuperabilidad permitiendo reiniciar módulos del sistema que se encuentren en un estado no contemplado en los requerimientos funcionales.
- **Seguridad:** En el atributo de calidad referente a disponibilidad debido a que actualizaciones del sistema no ocasionarán interrupciones de servicio para los usuarios.
- **Mantenimiento:** En los atributos de capacidad de prueba y comprensibilidad por medio del registro de servicios del Framework OSGi, que facilita el desarrollo de pruebas unitarias y de integración. Las pruebas unitarias pueden ser ejecutadas por módulos de prueba que acceden al registro de OSGi y validan los servicios publicados y las pruebas de integración por medio de composición de servicios. Adicionalmente en este tipo de requerimientos no funcionales fue mejorado el atributo de calidad referente a la estabilidad del sistema debido a que en la nueva plataforma es posible mantener en tiempo de ejecución diferentes versiones de los módulos del

sistema, supliendo dependencias que se encontrarían en conflicto en entornos solo soportados por la plataforma de Java, en donde se presentarían fallas en tiempo de ejecución a causa de incompatibilidad de versiones.

En la tabla 1 se listan las pruebas realizadas a las implementaciones desarrolladas en la práctica empresarial y en la tabla 2 son presentadas las pruebas al sistema CITOSOFT despues de la modularización.

Todas las pruebas diseñadas para verificar los requerimientos no funcionales que extienden el Framework Vaadin fueron documentadas para posteriores validaciones y adicionadas a la documentación del proyecto de CITOSOFT en Biosys Ltda.

TABLA 1. PRUEBAS NO FUNCIONALES ASOCIADAS A LA MODULARIZACION

ACCION	RESULTADO ESPERADO	RESULTADO ACTUAL
Publicar una aplicación Vaadin en el sistema de registro de OSGi.	El <i>Bundle</i> com.biosys.vaadinapplicationmanager debe detectar la aplicación y registrarla en el servidor de aplicaciones HTTP. La aplicación Vaadin debe ser visible en la web por medio de la URL previamente configurada.	√ La publicación de una aplicación Vaadin en el sistema de registro de OSGi fue validada exitosamente y publicada en el servidor de aplicaciones HTTP.
Publicar un servicio que implemente la interfaz UIService.	El administrador de interfaz del <i>Bundle</i> com.biosys.uicontroller debe detectar el nuevo servicio en ejecución y notificar a los contenedores dinámicos para su correspondiente renderizacion en la interfaz de los usuarios.	√ El servicio publicado fue registrado satisfactoriamente por el administrador de servicios de interfaz y los diferentes contenedores dinámicos de los usuarios que se encontraban ejecutando la aplicación fueron notificados y actualizados.
Verificar para un grupo de usuarios la ejecución	La interfaz de usuario debe responder independientemente a	√ Se verifico positivamente en el sistema que al haber varios usuarios ejecutando el mismo

concurrente de un servicio de tipo UIService.	cada una de las solicitudes sin afectar las transacciones de los otros usuarios.	servicio, la interfaz de usuario responde independientemente a cada una de las solicitudes sin afectar las transacciones de los otros usuarios.
Publicar diferentes versiones de un servicio que implemente la interfaz UIService	La interfaz para un grupo de usuarios debe responder de forma tardía o explícita de acuerdo al modelo de versionamiento.	√ Fue publicado el mismo servicio iniciando con la versión 1.0. Se actualizó a la versión 1.2 y los usuarios que se encontraban realizando transacciones basadas en ese servicio fueron mantenidos en la versión 1.0. Los usuarios que iniciaron sesión luego de la publicación de la versión 1.2 fueron direccionados a la versión más reciente. Posteriormente fue actualizado el servicio a la versión 2.0 lo que desencadenó interrupción de las operaciones de todos los usuarios asociados al servicio en las versiones 1.0 y 1.2, y fueron actualizados a la versión más reciente.
Seleccionar un servicio e interactuar concurrentemente	Todas las solicitudes vía HTTP y WebSockets generadas por el grupo	Todas las solicitudes realizadas vía HTTP y WebSockets por diferentes usuarios fueron

con diferentes usuarios y de esta manera desencadenar solicitudes de datos al servidor.	de usuarios deben ser notificadas al supervisor de solicitudes registrado en la aplicación vaadin.	interceptadas por el supervisor de solicitudes satisfactoriamente.
Instalar, remover, detener e iniciar algunos bundles del sistema por medio de la interfaz proporcionada por el <i>bundle</i> com.biosys. bundlemanager.	Verificar si la interfaz usuario publicada por el <i>Bundle</i> com.biosys. bundlemanager permite instalar, remover, detener e iniciar <i>Bundles</i> .	√ Por medio de la interfaz de usuario publicada por el Bundle com.biosys.bundlemanager se comprobó el ciclo de vida de diferentes <i>bundles</i> del sistema CITOSOFT.
Instalar, remover, detener e iniciar algunos Bundles del sistema por medio de la terminal del Framework OSGi.	Verificar si la interfaz de usuario publicada por el <i>Bundle</i> com.biosys. bundlemanager sincroniza los eventos sobre los <i>Bundles</i> del sistema y muestra a los usuarios el estado actual.	√ Se realizó la instalación de un servicio por medio de la consola del framework OSGi, y la interfaz en los usuarios que estaban haciendo seguimiento al ciclo de vida de los <i>bundles</i> fueron notificados. Posteriormente se detuvo un servicio y se inició el servicio que fue instalado en el sistema previamente. Las dos operaciones fueron realizadas y notificadas con éxito.

TABLA 2. PRUEBAS REALIZADAS A LOS REQUERIMIENTOS FUNCIONALES DEL SISTEMA CITOSOFT.

ACCION	RESULTADO ESPERADO	RESULTADO ACTUAL
Realizar inicio de sesión en la aplicación CITOSOFT con diferentes usuarios, estableciendo combinaciones de usuario y contraseña correctos e incorrectos.	<p>Para los casos de combinación de usuario y contraseña correcta el sistema debe actualizar la interfaz y mostrar al usuario el menú de navegación.</p> <p>Para los casos erróneos el sistema debe informar al usuario y recomendarle acerca de la opción de recordar contraseña.</p>	<p>√ El sistema mostro con éxito el inicio de sesión de un usuario en el sistema.</p> <p>√ El inicio de sesión de un usuario con datos erróneos en el sistema no fue exitoso, por tanto se prosigue a la opción de recordar la contraseña.</p>
Realizar registro de un usuario con rol enferme@ en el sistema	El sistema debe validar cada uno de los campos de información personal del usuario, indicar que deben ser adjuntados una copia del documento de identidad y de la tarjeta profesional que serán utilizados por el administrador del sistema para habilitar las diferentes configuraciones de acuerdo a su rol como profesional. Debe ser enviado un mensaje al correo electrónico del usuario	√ El sistema valido correctamente la información personal del usuario y envió notificación exitosa de la creación de la cuenta al respectivo correo del usuario.

	notificándole de la creación de la cuenta de usuario.	
Realizar inicio de sesión en la aplicación CITOSOFT con un usuario registrado en el sistema pero aun no habilitado por el administrador del sistema.	El sistema debe notificarle al usuario que su cuenta aún no ha sido habilitada por el administrador y que será enviada al correo electrónico registrado información referente al proceso de habilitación por parte del administrador del sistema.	√ En esta prueba la validación del sistema fue satisfactoria pues el usuario no pudo iniciar sesión porque aún no había sido habilitado por el administrador. Luego de la habilitación del administrador fue enviada al correo electrónico registrado la información referente al proceso de habilitación por parte del administrador del sistema.
Con una cuenta asociada al rol enfermero, realizar el registro de una solicitud de examen indicando la información del paciente (Nombres, Apellidos, Documento de identificación, entre otros) de acuerdo a los	El sistema al ingresar el tipo de documento y número de identificación de la paciente, valida si el paciente ya fue registrado en el sistema. En caso de ser afirmativo los otros campos de información básica del paciente son cargados automáticamente en la interfaz de usuario así como información asociada a su historia clínica como los antecedentes de procedimientos.	√ Los datos del paciente introducidos y su respectiva validación fue satisfactoria en el sistema.

requerimientos funcionales RF3 y RF4 de la fase de pre-análisis.	El sistema luego de validar toda la información del paciente diligenciada por el enfermero debe hacer la correspondiente remisión de la muestra para el laboratorio o centro de diagnóstico seleccionado.	√ El sistema realizo exitosamente la remisión de la muestra para el laboratorio o centro de diagnóstico seleccionado.
Con una cuenta asociada al rol enfermero verificar el estado de solicitudes de examen de acuerdo a los requerimientos funcionales RF5 y RF6	<p>El sistema debe listar las citologías dando prioridad a las citologías con resultados anormales diagnosticadas recientemente. La lista debe estar ordenada por fecha de diagnóstico (Descendente) y resultado diagnostico (Anormal, Normal). Para las citologías diagnosticadas debe permitir al enfermero acceder al diagnóstico emitido por el cito-tecnólogo o medico patólogo en formato pdf para facilitar su impresión.</p> <p>El sistema debe permitir la búsqueda de citologías por medio del número de documento de la paciente o por resultado diagnóstico. Luego de realizada una búsqueda, el sistema de listar los nombres de las mujeres, cedula de ciudadanía, teléfono de contacto, código de la citología y el estado en el</p>	<p>√ El sistema lista correctamente las citologías dando prioridad a las citologías con resultados anormales y las ordena por fecha de diagnóstico.</p> <p>√ El sistema verifico exitosamente la búsqueda de citologías por medio de las dos opciones (número de documento de la paciente o por el resultado diagnostico).</p>

	que se encuentra la citología.	
Con una cuenta asociada al rol citotecnologo radicar una citología de acuerdo al requerimiento funcional RF7	El sistema debe listar las citologías sin analizar ordenadas por fecha de toma de la muestra en orden descendente. Luego de seleccionar una citología es asignado un número único de radicación de la muestra que permite determinar su ubicación en el inventario físico para posteriores análisis.	✓ La prueba que se hizo al sistema fue realizada satisfactoriamente, se listaron correctamente las citologías sin analizar, ordenadas por orden de fecha de toma de la muestra en orden descendente. En seguida se seleccionó una citología y se comprobó que esta tenía un número único de radicación de la muestra la cual permitió su ubicación en el inventario físico para posteriores análisis.
Digitalizar una lámina de citología de acuerdo a los requerimientos funcionales RF8, RF9, RF10, RF11, RF12	<p>El sistema debe desplegar una interfaz de digitalización que detecte el microscopio del citotecnologo y permita realizar el proceso de digitalización.</p> <p>Al mismo tiempo el sistema debe permitir al citotecnologo adicionar hallazgos al diagnóstico de acuerdo a la clasificación Bethesda.</p>	✓ Actualmente la interfaz de digitalicen permite la digitalización de las láminas de citología. Sin embargo deben ser realizadas mejoras al proceso pues no cumple con los atributos de calidad AC1 y AC4. El tiempo de digitalización de una muestra de citología a 40X es de 40 minutos.

	Luego de digitalizada la muestra y emitido el diagnostico; la citología de cambiar a estado diagnosticada. En el caso que la citología requiera un segundo análisis por parte de un medico patólogo la citología debe ser remitida y cambiar su estado a espera de diagnóstico de medico patólogo	
Con una cuenta asociada al rol medico patólogo emitir el diagnostico de una citología de acuerdo a los requerimientos funcionales RF15, RF16 y RF17	El sistema debe listar las citologías sin analizar ordenadas por fecha de toma de la muestra en orden descendente. Luego de seleccionar una citología es asignado un número único de radicación y emitido el diagnostico por parte del rol patólogo. En la interfaz es posible observar los hallazgos registrados por el citotecnologo, los cuales se pueden editar de acuerdo a nuevos hallazgos encontrados así como eliminar en los casos donde la interpretación fue errónea.	<p>√ El sistema listo satisfactoriamente las citologías sin analizar mostrando que están ordenadas por fecha de toma de la muestra en orden descendente.</p> <p>√ Se seleccionó una citología y verificó que fue asignado un número único de radicación de la muestra.</p> <p>√ Fue emitido el diagnostico con el rol citotecnologo y se ver verificó con rol enfermero que el resultado estaba disponible para impresión.</p>

CONCLUSIONES

La modalidad de práctica empresarial ayuda en el desarrollo profesional de los estudiantes de la Escuela de Ingeniería de Sistemas de la Universidad Industrial de Santander, complementando el proceso de formación académica por medio de la aplicación de los conocimientos adquiridos a problemas de la industria y la sociedad.

La modalidad de práctica empresarial permite profundizar en tecnologías utilizadas en la industria, en donde por medio de acompañamiento y capacitación es posible obtener certificaciones internacionales en tecnologías que enriquecen el perfil profesional.

El Framework OSGi es una infraestructura eficiente que facilita la administración y desarrollo de aplicaciones empresariales basadas en servicios, proporcionando herramientas que incrementan la escalabilidad, portabilidad, modularidad y dinamismo de las aplicaciones por medio de un conjunto de especificaciones que continuamente se adaptan a las necesidades de la industria.

El Framework Vaadin optimiza el desarrollo de aplicaciones web por medio del modelo orientado a servidor que controla las comunicaciones con los clientes sobre AJAX y renderiza interfaces enriquecidas basadas en el Google Web Toolkit GWT y HTML5.

La extensión del Framework Vaadin propuesta en la práctica empresarial proporciona una arquitectura modular que combinada con las características del Framework Vaadin simplifica la fase de mantenimiento de software, permitiendo actualizar componentes individuales sin afectar su funcionamiento global de las aplicaciones.

BIBLIOGRAFIA

ANÁLISIS Y DISEÑO DE UN SISTEMA DE TELECITOLOGIA QUE APOYE EL PROGRAMA DE DETECCIÓN TEMPRANA DE CANCER CERVICOUTERINO EN COLOMBIA [RECURSO ELECTRÓNICO] / LINA MARIA GARCES R.; DIRECTORES LOLA XIOMARA BAUTISTA ROZO, OLGA M., ALVAREZ O.

B. Bruegge and A. H. Dutoit, Object-Oriented Software Engineering. Using UML, Patterns and Java. Second Edition. Pearson Prentice Hall, 2004.

Cummins, Holly, and Timothy Ward. *Enterprise OSGi in Action: With Examples Using Apache Aries*. Manning Publications Co., 2013.

D. Mairiza, D. Zowghi, and N. Nurmuliani, "An investigation into the notion of non-functional requirements," in Proceedings of the 2010 ACM Symposium on Applied Computing, SAC '10, (New York, NY, USA), pp. 311–317, ACM, 2010.

D. Miranda, J. Barrero, and J. Echeverri, Estudio para la detección temprana del cáncer de cuello uterino. Editorial Sic. Universidad Industrial de Santander., 2007.

D. Palacio, "Plan nacional para el control del cáncer en colombia 2010-2019.." Instituto Nacional de Cancerología. Empresa Social del Estado.(2010).

Duarte, Alejandro. *Vaadin 7 UI Design by Example: Beginner's Guide*. Packt Publishing Ltd, 2013.

[En línea] Disponible en: <http://www.osgi.org/Technology/WhyOSGi>. Visitado Junio de 2013.

[En línea] Disponible en: <http://docs.oracle.com/javaee/5/tutorial/doc/bnaby.html#indexterm-47> Visitado Junio de 2014

[En línea] Disponible en: <http://openjdk.java.net/jeps/201> Visitado Junio de 2014

[En línea] Disponible en: <http://openjdk.java.net/jeps/200> Visitado Junio de 2014

Instituto Nacional de Cancerología-Instituto Geográfico Agustín Codazzi. Atlas de la mortalidad por cáncer en Colombia. ; 2010.

MALDONADO García Emilio José, SERVICIO DE INTERFAZ DE USUARIO SOBRE EL FRAMEWORK DE OSGi, PDF, Paginas (1 - 19), Disponible en <http://bibing.us.es/proyectos/abreproy/11033/fichero/Volumen1%252Fcapitulo1.pdf>

Manual de Identidad Corporativa de Biosys Ltda.

MOTA Porta Julio Angel, UPnP MEDIA RENDERER PARA PLATAFORMA OSGi, Proyecto fin de carrera, Julio de 2009.

Murillo R. Control del cáncer de cuello uterino en Colombia: triunfos y desafíos de la tamización basada en la citología cérvico-uterina. Biomédica. 2008;28:468-70.

PIRCENO Juan Manuel y TENZER Ken, API DE ALTO NIVEL GENERICO PARA DESARROLLO DE APLICACIONES DE DOMOTICA, Informe de Proyecto, Montevideo – Uruguay, Febrero 26 de 2010.

RICHARD Kaegi Simon, Deugo Dwight, MODULAR JAVA WEB APPLICATIONS.

Roberto Montero Miguel, Aprendiendo OSGI mediante Ejemplos [En línea]
Disponible en: <https://www.scribd.com/doc/46748336/OSGI-Roberto-Montero>.
Visitado Mayo de 2014

S. Hall, PAULS Karl, McCulloch Stuart y SAVAGE David, OSGi IN ACTION
CREATING MODULAR APPLICATIONS IN JAVA, In IEEE Consumer
Communications and Networking Conference 2004.

VALENCIA, JAIME ALBERTO FLAUTERO, and FABIO ANDRES SÁNCHEZ
BAPTISTE. "ESTUDIO Y EVALUACIÓN DE LA ESPECIFICACIÓN OSGI R3
PARA ARQUITECTURAS MÓVILES."

World Health Organization. International Agency for Research on Cancer-
GLOBOCAN. [Online]; 2008 [cited 2012 septiembre 13. Available from:
<http://globocan.iarc.fr/factsheets/cancers/cervix.asp>.

CITAS BIBLIOGRAFICAS

- [1] D. Miranda, J. Barrero, and J. Echeverri, Estudio para la detección temprana del cáncer de cuello uterino. Editorial Sic. Universidad Industrial de Santander., 2007.
- [2] D. Palacio, "Plan nacional para el control del cáncer en Colombia 2010-2019.." Instituto Nacional de Cancerología. Empresa Social del Estado.(2010).
- [3] ANÁLISIS Y DISEÑO DE UN SISTEMA DE TELECITOLOGIA QUE APOYE EL PROGRAMA DE DETECCIÓN TEMPRANA DE CANCER CERVICOUTERINO EN COLOMBIA [RECURSO ELECTRÓNICO] / LINA MARIA GARCES R.; DIRECTORES LOLA XIOMARA BAUTISTA ROZO, OLGA M., ALVAREZ O.
- [4] Manual de Identidad Corporativa de Biosys Ltda.
- [5] S. Hall, PAULS Karl, McCulloch Stuart y SAVAGE David, OSGi IN ACTION CREATING MODULAR APPLICATIONS IN JAVA, In IEEE Consumer Communications and Networking Conference 2004.
- [6] VALENCIA, JAIME ALBERTO FLAUTERO, and FABIO ANDRES SÁNCHEZ BAPTISTE. "ESTUDIO Y EVALUACIÓN DE LA ESPECIFICACIÓN OSGI R3 PARA ARQUITECTURAS MÓVILES."
- [7] MALDONADO Garcia Emilio Jose, SERVICIO DE INTERFAZ DE USUARIO SOBRE EL FRAMEWORK DE OSGi, PDF, Paginas (1 - 19), Disponible en <http://bibing.us.es/proyectos/abreproy/11033/fichero/Volumen1%252Fcapitulo1.pdf>

[8] Roberto Montero Miguel, Aprendiendo OSGI mediante Ejemplos [En linea] Disponible en: <https://www.scribd.com/doc/46748336/OSGI-Roberto-Montero>. Visitado Mayo de 2014

[9] MOTA Porta Julio Angel, UPnP MEDIA RENDERER PARA PLATAFORMA OSGi, Proyecto fin de carrera, Julio de 2009.

[10] PIRCENO Juan Manuel y TENZER Ken, API DE ALTO NIVEL GENERICO PARA DESARROLLO DE APLICACIONES DE DOMOTICA, Informe de Proyecto, Montevideo – Uruguay, Febrero 26 de 2010.

[11] RICHARD Kaegi Simon, Deugo Dwight, MODULAR JAVA WEB APPLICATIONS.

[12] [En linea] Disponible en: <http://www.osgi.org/Technology/WhyOSGi>. Visitado Junio de 2013.

[13] Duarte, Alejandro. *Vaadin 7 UI Design by Example: Beginner's Guide*. Packt Publishing Ltd, 2013.

[14] [En linea] Disponible en: <http://docs.oracle.com/javase/5/tutorial/doc/bnaby.html#indexterm-47> Visitado Junio de 2014

[15] [En linea] Disponible en: <http://openjdk.java.net/jeps/201> Visitado Junio de 2014

[16] [En linea] Disponible en: <http://openjdk.java.net/jeps/200> Visitado Junio de 2014

[17] Cummins, Holly, and Timothy Ward. *Enterprise OSGi in Action: With Examples Using Apache Aries*. Manning Publications Co., 2013.

- [18] World Health Organization. International Agency for Research on Cancer- GLOBOCAN. [Online]; 2008 [cited 2012 septiembre 13. Available from: <http://globocan.iarc.fr/factsheets/cancers/cervix.asp>.
- [19] Instituto Nacional de Cancerología-Instituto Geográfico Agustín Codazzi. Atlas de la mortalidad por cáncer en Colombia. ; 2010.
- [20] Murillo R. Control del cáncer de cuello uterino en Colombia: triunfos y desafíos de la tamización basada en la citología cérvico-uterina. Biomédica. 2008;28:468-70.
- [21] B. Bruegge and A. H. Dutoit, Object-Oriented Software Engineering. Using UML, Patterns and Java. Second Edition. Pearson Prentice Hall, 2004.
- [22] D. Mairiza, D. Zowghi, and N. Nurmuliani, "An investigation into the notion of non-functional requirements," in Proceedings of the 2010 ACM Symposium on Applied Computing, SAC '10, (New York, NY, USA), pp. 311–317, ACM, 2010.