

Universidad de las Ciencias Informáticas

Facultad 3



Título: Sistema de Apoyo a la Vigilancia Tecnológica en el Centro de Gobierno Electrónico (CEGEL).

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

Autor (es): Oniel Bicet Castañeda
Rafael Romero Cáceres

Tutor (es): Msc. Yarina Amoroso Fernández
Ing. Daniel Varona Cordero

La Habana, junio del 2014
“Año 56 de la Revolución”



El hombre puede hacer de sí mismo muchas cosas producto de su propio esfuerzo físico y espiritual, el que se proponga cultivar la virtud la cultiva, el que se proponga alcanzar los más altos niveles de conocimiento los alcanza.

Fidel Castro Ruz

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año 2014.

Autor: Oniel Bicet Castañeda

Autor: Rafael Romero Cáceres

Tutor: MsC. Yarina Amoroso Fernández

Tutor: Ing. Daniel Varona Cordero

DEDICATORIA

A mi familia, especialmente a mis padres, mi hermana y mi abuela Margarita.

Oniel

A mis padres, Marlén y Rafael, por todo lo que han hecho para que yo pudiera alcanzar esta meta.

A mi hermana Darlén, por todo el amor que me brinda siempre.

Rafael

AGRADECIMIENTOS

Quiero agradecer a mi familia, especialmente a mis padres y a mi hermana por siempre confiar en mí y apoyarme en todos los momentos de mi vida.

Gracias a todos los amigos que me han acompañado a lo largo de mi vida.

Gracias a Lily, Oscarlos, Rinat, Amarilis, Arlety, e Ibelise por haber compartido conmigo estos difíciles 5 años en la universidad y por su apoyo incondicional.

Gracias a las amistades que por un motivo u otro, no se encuentran presentes para compartir este momento conmigo, en especial a Mavis.

Gracias a los tutores por toda la ayuda que me han brindado.

Gracias a mi compañero de tesis por su compromiso y dedicación.

Gracias a todas las personas que de una forma u otra me han ayudado a materializar este sueño.

Oniel

AGRADECIMIENTOS

Quiero Agradecer...

Y de forma especial, a mis padres por ser las personas más especiales en mi vida, por ser mis mejores amigos, por todo ese amor incondicional que siempre tienen para mí, por ser el más claro ejemplo de sacrificio y dedicación que he visto. Por educarme y tratar de guiarme siempre. Por tantas cosas que el espacio es pequeño para describirlas. Gracias a ellos estoy aquí y soy quien soy.

A mi hermana, por ser mi amiga, por permitirme quererla y guiarla siempre, por estar ahí en todos los momentos buenos y malos de mi vida.

A tía Belkís, por ser mi segunda madre, a mis tíos Yakelín, Mary, Pipo, Siuleima, Gustavo, Rolando, Jorge, en fin a mi familia de forma general por ser la familia más unida que he conocido jamás.

A los tutores Yarina y Daniel, por siempre guiarnos y ayudarnos a mejorar, y por dejarnos ocupar su limitado tiempo.

A Oniel, mi compañero de Tesis, por comprometerse con esta investigación y dedicarle todo el tiempo posible.

Al tribunal por ayudarnos a perfeccionar la investigación, de manera especial a las profes Diana y Yoenia.

A mis amigos y compañeros de grupo, los que se encuentran hoy y los que por algún motivo están distantes.

A la universidad y todos los que han sido mis profesores, por darme la oportunidad y enseñarme a ser mejor cada día.

A todos los que de alguna u otra forma han contribuido con mi formación y con este resultado.

Rafael

RESUMEN

La información es considerada un recurso de significativa importancia para las organizaciones, se ha convertido en un factor estratégico y de gran valor para el buen desempeño de las mismas. Entre las prácticas más utilizadas para la adquisición y gestión de este recurso se encuentra la Vigilancia Tecnológica.

La presente investigación tiene como objetivo desarrollar un sistema que permita el apoyo a la Vigilancia Tecnológica para de esta forma eliminar carencias en el proceso de Vigilancia Tecnológica que dificultaban la disponibilidad de información actualizada relacionada con los productos que se desarrollan, de interés para los gestores de proyectos y su equipo de trabajo en el Centro de Gobierno Electrónico (CEGEL). Para ello se realizó un estudio sobre: conceptos asociados a Vigilancia Tecnológica, Sistemas de Vigilancia Tecnológica y las funcionalidades de Sistemas de Vigilancia Tecnológica similares existentes a nivel nacional e internacional. Además se analizaron las características de las herramientas seleccionadas para el desarrollo de la propuesta de solución y se pusieron en práctica técnicas de la Ingeniería de Software basadas en la metodología XP para una mejor asimilación del negocio. Como resultado fundamental se obtuvo el Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL haciendo uso del Sistema de Gestión de Contenidos (CMS) Drupal el cual brinda diversas funcionalidades y facilidades para un sistema de este tipo.

Palabras claves: Sistemas de Vigilancia Tecnológica, Sistema de Gestión de Contenido, Gestión de Información, Drupal.

ÍNDICE

DECLARACIÓN DE AUTORÍA.....	3
DEDICATORIA.....	4
AGRADECIMIENTOS.....	5
AGRADECIMIENTOS.....	6
RESUMEN.....	7
ÍNDICE.....	8
ÍNDICE DE TABLAS.....	10
ÍNDICE DE IMÁGENES.....	11
INTRODUCCIÓN.....	12
CAPÍTULO 1: FUNDAMENTOS TEÓRICOS SOBRE EL DESARROLLO DEL SISTEMA DE APOYO A LA VIGILANCIA TECNOLÓGICA EN EL CENTRO DE GOBIERNO ELECTRÓNICO.....	15
1.1 Vigilancia Tecnológica.....	15
1.2 Sistemas de Vigilancia Tecnológica.....	18
1.3 Tendencias Actuales de los Sistemas de Vigilancia Tecnológica a Nivel Internacional y Nacional.....	19
1.4 Sistemas de Gestión de Contenidos (CMS).....	25
1.5 Metodologías de Desarrollo de Software.....	29
1.6 Lenguajes para el Desarrollo.....	34
1.7 Herramientas CASE.....	35
1.8 Entorno de Desarrollo Integrado (IDE).....	36
1.9 Servidor Web.....	37
1.10 Sistemas Gestores de Bases de Datos (SGBD).....	37
1.11 Conclusiones Parciales.....	39
CAPÍTULO 2: PROPUESTA DEL SISTEMA DE APOYO A LA VIGILANCIA TECNOLÓGICA EN EL CENTRO DE GOBIERNO ELECTRÓNICO (CEGEL).....	40

2.1 Descripción del Sistema.....	40
2.2 Modelo de Dominio.....	41
2.3 Especificación de Requisitos del Sistema.....	43
2.4 Fase de Exploración	47
2.5 Planificación de la Entrega	49
2.6 Diseño del Sistema Propuesto	52
2.7. Conclusiones Parciales	56
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA DE APOYO A LA VIGILANCIA TECNOLÓGICA EN EL CENTRO DE GOBIERNO ELECTRÓNICO (CEGEL).	57
3.1 Estudio Detallado de la Técnica de Programación Utilizada para el Desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL	57
3.2 Estándares de Codificación.....	60
3.3 Tareas de Programación	66
3.4 Diagrama de Despliegue	67
3.5 Estrategia de Prueba	67
3.6 Conclusiones Parciales	71
CONCLUSIONES GENERALES.....	73
RECOMENDACIONES.....	74
BIBLIOGRAFÍA	75
GLOSARIO	79

ÍNDICE DE TABLAS

Tabla 1: Comparación de los Sistemas Analizados.....	24
Tabla 2: Comparación entre Metodologías Tradicionales y Ágiles	30
Tabla 3: Requisitos Funcionales	43
Tabla 4: Historia de Usuario Nro. 1	48
Tabla 5: Historia de Usuario Nro. 2	48
Tabla 6: Estimación de esfuerzo por Historias de Usuario	49
Tabla 7: Plan de Iteraciones	51
Tabla 8: Plan de Entregas	51
Tabla 9: Módulo-Responsabilidad-Colaboración	64
Tabla 10: Tarea de Programación #1	66
Tabla 11: Tarea de Programación #2.....	66
Tabla 12: Caso de Prueba de Aceptación HU4_P1	69
Tabla 13: Caso de Prueba de Aceptación HU4_P2.....	69

ÍNDICE DE IMÁGENES

Figura 1: Modelo de Dominio.....	42
Figura 2: Arquitectura del Sistema	53
Figura 3: Ejemplo de Indentación en Arrays	62
Figura 4: Diagrama de Despliegue	67
Figura 5: Resultados de las Pruebas de Aceptación	70
Figura 6: Resultados de las Pruebas de Usabilidad	71

INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones (TICs) se han convertido, a una gran velocidad, en parte importante de nuestras vidas. Actualmente con los avances de las TICs se han logrado hacer cosas que el ser humano consideraba imposible llevar a cabo, se han implementado software y hardware, en áreas como la educación, salud, empresas, infraestructuras, entre otras. La información impulsada por el vertiginoso avance científico en un marco socioeconómico y sustentado por el uso generalizado de las potentes y versátiles tecnologías de la información y las comunicaciones conlleva cambios que alcanzan todos los ámbitos de la actividad humana. Sus efectos se manifiestan de manera muy especial en las actividades laborales y en el mundo educativo, donde todo debe ser revisado, desde las instituciones educativas hasta la formación básica precisan las personas, la forma de enseñar y de aprender, las infraestructuras y los medios que se utilizan para ello. Cada día la información se hace más diversa en temas específicos y en consecuencia los mecanismos de retroalimentación deben ser reajustados constantemente.

Por ello, las organizaciones despliegan una serie de acciones encaminadas a garantizar una adecuada actualización sobre información muy específica, relacionada con los procesos que ejecuta y los productos que desarrolla, de manera que les garantice el éxito en medio de la llamada Sociedad de la Información.

Esto permite a la organización estar en condiciones de conocer el escenario donde debe desarrollar su estrategia, o sea, sus amenazas y oportunidades. A mayor grado de comunicación entre la organización y su entorno, más posibilidades para generar, desarrollar y asimilar nuevas capacidades tecnológicas. En este contexto surge el término de Vigilancia Tecnológica.

La Vigilancia Tecnológica consiste en un proceso organizado, selectivo y permanente, de captar información interna y externa de la propia organización sobre ciencia y tecnología; seleccionarla, analizarla, difundirla y comunicarla, para convertirla en conocimiento para tomar decisiones con menor riesgo y poder anticiparse a los cambios (Angelozzi, y otros, 2011).

Un Sistema de Vigilancia Tecnológica puede definirse como la búsqueda, detección, análisis y comunicación a los directivos de la empresa de informaciones orientadas a la toma de decisiones sobre amenazas y oportunidades externas en el ámbito de la ciencia y la tecnología. La Vigilancia Tecnológica constituye un factor clave para la gestión de proyectos de I+D+i. Existe un gran cúmulo de información disponible en distintas fuentes (artículos científicos-técnicos, patentes, bases de datos, buscadores especializados), pero si no se vigila y detecta correctamente puede que una organización termine investigando lo ya investigado o que quiera patentar lo ya patentado (Angelozzi, y otros, 2011).

Además un Sistema de Vigilancia Tecnológica es, una herramienta de carácter colaborativo, donde se hace necesario el acceso de todos los miembros a una determinada plataforma para introducir información de interés organizacional, así como para consultar documentos (Negrín, 2012).

Actualmente en la Universidad de las Ciencias Informáticas (UCI), específicamente en el Centro de Gobierno Electrónico (CEGEL) existe un grupo de trabajo dedicado a temas relacionados con la Vigilancia Tecnológica. El trabajo del grupo se reduce a divulgar las políticas de Vigilancia Tecnológica determinadas por un grupo homónimo a nivel UCI; así como actualizar a ese grupo central las líneas de investigación del CEGEL como vía de actualización del(los) objeto(s) de vigilancia. En consecuencia existen retrasos en la retroalimentación que tiene el CEGEL y sus gestores de proyectos sobre los temas relacionados con sus intereses de desarrollo/comercialización, últimas herramientas en el mercado y espacios para divulgar su producción científica.

De lo anterior se hace necesario que el grupo de Vigilancia Tecnológica del CEGEL cuente con métodos y herramientas que le permitan cobrar mayor protagonismo monitoreando el entorno externo e interno del Centro, en el afán de proveer a sus gestores de proyectos y equipos de trabajo en general, de información actualizada relacionados con sus intereses específicos.

Por todo lo antes expuesto se identifica como **problema científico**:

Las carencias en el actual proceso de Vigilancia Tecnológica en el CEGEL dificultan la disponibilidad de información actualizada relacionada con los productos que se desarrollan en el centro, de interés para los gestores de proyectos y su equipo de trabajo.

Se tiene como **objeto de estudio**: el proceso de Vigilancia Tecnológica. La investigación se desarrolla bajo el siguiente **campo de acción**: los sistemas web para Vigilancia Tecnológica.

Para dar solución a la problemática planteada se establece como **objetivo general**, desarrollar un sistema web de apoyo a la Vigilancia Tecnológica que permita a los gestores de proyectos y su equipo de trabajo, tener disponible información actualizada relacionada con los productos que se desarrollan en el CEGEL; y como **objetivos específicos**:

1. Construir los referentes teóricos relacionando los aspectos fundamentales del proceso de Vigilancia Tecnológica que sustentan la investigación, mediante los cuales se consulta, extrae y recopila información relevante sobre el problema y su contextualización.
2. Diseñar un sistema web de soporte al proceso de Vigilancia Tecnológica en el CEGEL.
3. Validar la propuesta de solución.

La investigación se sostiene con la siguiente **idea a defender**:

Contar con un Sistema de Apoyo a la Vigilancia Tecnológica permitirá a los gestores de proyecto y su equipo de trabajo en el CEGEL tener disponible información actualizada relacionada con los productos que se desarrollan en el centro.

De los **métodos teóricos** se utilizaron:

Histórico – lógico: para determinar la evolución y desarrollo de los Sistemas de Vigilancia Tecnológica, desde su surgimiento hasta la actualidad y las tendencias actuales de estos sistemas.

Analítico Sintético: para identificar, analizar y seleccionar los conceptos, las definiciones y las teorías más importantes relacionados con los Sistemas de Vigilancia Tecnológica que permita luego generar una propuesta adecuada a la situación planteada.

Modelación: para representar de una manera lógica los procesos de los Sistemas de Vigilancia Tecnológica que se automatizarán, mediante modelos y diagramas que ayuden a comprender el objeto en su totalidad.

De los **métodos empíricos** se empleó:

Entrevista: para identificar los requisitos funcionales y no funcionales del sistema.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS SOBRE EL DESARROLLO DEL SISTEMA DE APOYO A LA VIGILANCIA TECNOLÓGICA EN EL CENTRO DE GOBIERNO ELECTRÓNICO.

En este capítulo se realiza un estudio del estado del arte de los Sistemas de Vigilancia Tecnológica, todo lo relacionado con estos, sus principales características y componentes, con el objetivo de encontrar características comunes que se puedan aplicar en la solución. Se estudian las metodologías y las herramientas que pueden ser utilizadas en el desarrollo del sistema.

1.1 Vigilancia Tecnológica

La Vigilancia Tecnológica (VT) es un proceso de carácter informativo/documental selectivo que recopila y organiza información y documentos sobre un área de especialización muy concreta y que está dirigido específicamente a un grupo de usuarios (Fernandez Fuentes, y otros, 2009).

Por su parte la norma UNE 166000: 2006 sobre terminología y definiciones de las actividades de I+D+i (Investigación + Desarrollo + innovación tecnológica) para Sistemas de Vigilancia Tecnológica e Inteligencia Competitiva se expone la Vigilancia Tecnológica como un proceso organizado, selectivo y permanente, de captar información del exterior y de la propia organización sobre ciencia y tecnología, seleccionarla, analizarla, difundirla y comunicarla, para convertirla en conocimiento para tomar decisiones con menor riesgo y poder anticiparse a los cambios (AENOR, 2011).

En consecuencia la información científico-tecnológica que resulta de este proceso sirve de apoyo a la toma de decisiones, en especial en las empresas de base tecnológica y en las instituciones de investigación y desarrollo. Por su carácter continuo es un proceso movido por la exigencia por disponer de información crítica en su afán por mejorar la competitividad y al aumento de la efectividad de las organizaciones en contextos innovadores. Esta genera un valor agregado, partiendo de la obtención de datos y/o información para una adecuada toma de decisión, y finalizando en un proceso de obtención de conocimientos que termina en la inteligencia competitiva al decir de (Angelozzi, y otros, 2011).

A partir de lo analizado se puede puntualizar la VT como:

- Conocer las tecnologías emergentes que se están investigando, publicando o patentando en determinada área.
- Conocer la dinámica de las tecnologías, es decir, qué tecnologías se están imponiendo y cuáles quedaron obsoletas.
- Conocer las líneas de investigación y la trayectoria tecnológica llevadas a cabo por las principales empresas que compiten en el área.

- Reducción de riesgos de toma de decisiones, al conocer mejor donde posicionarse con nuestras estrategias.

Una vez enmarcado el término VT en las particularidades de la presente investigación se hace necesario desagregarlo en los distintos servicios que presta con el objetivo de identificar el tratamiento que pudiera tener la propuesta de solución a partir de este análisis.

Servicios que ofrece la Vigilancia Tecnológica

La VT supone una oportunidad al trasladar los métodos, técnicas y productos propios de la información y documentación científica al ámbito de los sistemas de gestión de las organizaciones. La mayoría de los servicios ofrecidos por la VT utilizan las herramientas típicas del trabajo documental realizado por los profesionales de la información en bibliotecas y centros de documentación.

La bibliografía consultada expone, entre los servicios apropiados para la Vigilancia Tecnológica, los siguientes (Angelozzi, y otros, 2011) (Parada Soto, y otros, 2005):

- **Alertas informativas**
- **Compilaciones de noticias e informaciones científicas y técnicas**
- **Búsqueda de información especializada**
- **Obtención de artículos científicos y técnicos**
- Agenda de eventos (talleres, jornadas, seminario) en un tema dado
- Lista de discusión
- Mapas de fuentes de información
- Monitoreo de información
- Productos informativos de valor agregado
- Estudios de tendencias

Se resaltan aquellos servicios que estarán comprendidos dentro de la propuesta de solución dada su importancia para el CEGEL. Luego se procede a presentar los distintos tipos de VT con el objetivo de identificar el propio, que viene determinado por los servicios ya seleccionados.

Tipos de Vigilancia Tecnológica

Las necesidades de información surgen de las organizaciones y pueden ser muy diversas: necesidad de conocer tendencias en los mercados, clientes, productos, tecnologías y el entorno.

Atendiendo a esto los tipos de VT se pueden clasificar como sigue acorde a su objeto de vigilancia:

- **VT de tecnologías:** Este tipo de vigilancia va a permitir conocer cuáles son las tecnologías disponibles y/o emergentes, los países líderes en estas tecnologías, las nuevas líneas de investigación en un sector/área, las tecnologías que emplean los competidores y las patentes solicitadas y concedidas en el sector/área.
- **VT de mercados:** Para realizar estudios de mercado, posicionamiento en el mercado, cuota de mercado, distribuidores, canales de distribución, evolución de las necesidades de los clientes y los mercados de alto valor.
- **VT de productos:** Permite dar seguimiento de los productos de la competencia y estar al tanto del desarrollo de nuevos productos.
- **VT del entorno:** Dirigido a la reglamentación, leyes y normas técnicas, factores externos tales como políticas, sociología, medioambiente y economía.
- **VT de clientes:** Este tipo de vigilancia está centrado en el perfil de los clientes y el estudio de sus necesidades, demandas y hábitos.
- **VT de competidores:** Competidores actuales y potenciales. Objetivos y estrategias de los competidores.

Resultando ser la VT de tecnologías el tipo más apropiado en la clasificación de la solución que se propone; sin desconocer la importancia del resto de los objetos de vigilancia, que por limitaciones de tiempo y concreción de la actual investigación se recomiendan en posteriores versiones de la misma.

Expuestos los términos, los servicios y su clasificación se procede a plantear la importancia implícita del propio proceso de VT para los procesos que soporta.

Importancia de la Vigilancia Tecnológica

El proceso de Vigilancia Tecnológica adquiere gran importancia en la toma de decisiones de una institución pues facilita, mediante procesos de captura y análisis, que la información sea utilizada en beneficio de las empresas o instituciones y viabiliza el hecho de estar atentos a muchos aspectos y señales claves que permiten anticipar el devenir y los cambios.

La vigilancia es considerada como una de las seis funciones claves para la gestión de la tecnología y la innovación en la actualidad. Su práctica proporciona las funcionalidades siguientes (Gestiopolis, 2014):

- Detectar las oportunidades antes que la competencia
- Conocer el estado del arte en su dominio empresarial
- Tomar posición en su sector
- Orientar a la I+D (Investigación + Desarrollo)
- Encontrar socios tecnológicos y financieros

Además existen otras razones por las que una empresa debe practicar la vigilancia:

- Para detectar los cambios del entorno, las nuevas tecnologías, mercados y competidores
- Para prevenir riesgos con respecto a patentes, productos, reglamentaciones, alianzas y nuevas inversiones
- Para detectar ideas y nuevas soluciones en I+D
- Para conocer nuevos socios, clientes y expertos

Lo anterior determina la orquestación del proceso en sistemas como parte del mapa funcional de las organizaciones, se hace necesario entonces describir las características principales de esta como sistema organizacional.

1.2 Sistemas de Vigilancia Tecnológica

La norma UNE 166006:2011 define los Sistemas de Vigilancia Tecnológica como la “parte del sistema general de gestión de la organización que comprende el conjunto de medios y recursos mediante los cuales, a partir de una cultura innovadora, se realiza la Vigilancia Tecnológica” (AENOR, 2011).

También puede definirse como la búsqueda, detección, análisis y comunicación a los directivos de la empresa de informaciones orientadas a la toma de decisiones sobre amenazas y oportunidades externas en el ámbito de la ciencia y la tecnología.

Un Sistema de Vigilancia Tecnológica es un proceso que se lleva a cabo de forma automatizada y que le permite a los miembros del sistema interactuar de manera dinámica, donde la información que se maneja es confidencial y clasificada, lo cual contribuye a la competitividad y al uso comercial de la empresa, dependiendo de sus características y sus propias necesidades. Son sistemas colaborativos, los cuales se componen de una estructura organizativa y un modelo de gestión, la planificación de actividades de Vigilancia Tecnológica, procedimientos, recursos humanos y componentes tecnológicos que permitan el flujo de información y el flujo de decisión. Los mismos evitan las pérdidas de tiempo, dinero, recursos y esfuerzos ya que permiten la elaboración adecuada de estrategias competitivas.

Características de los sistemas informáticos que brindan apoyo al proceso de Vigilancia Tecnológica.

- Muchos de estos entornos que gestionan el ciclo completo del proceso de VT integran conceptos avanzados de redes sociales, como es el caso de las plataformas Madri+d y Vigiale; permitiendo la interacción entre los usuarios a través de los foros o espacios de colaboración para la discusión de temas de interés.

- La arquitectura está diseñada para que facilite la interacción del usuario con la información, en forma amigable y personalizada.
- Destacan lo más relevante, mediante boletines informativos generados a partir de fuentes preseleccionadas, o le ofrecen a los profesionales suscribirse a los canales RSS a través de los cuales podrán tener acceso a la información.
- Tratamiento de la información descargada desde la Web (documentos en formato Word, HTML y PDF).
- Cuenta con buscadores que permiten detectar tanto la información interna como externa (información en la Web) de una organización.

1.3 Tendencias Actuales de los Sistemas de Vigilancia Tecnológica a Nivel Internacional y Nacional

A nivel internacional, los primeros países en implementar servicios de Vigilancia Tecnológica fueron Estados Unidos, Canadá y Francia. España es uno de los países de habla hispana que más experiencia posee en servicios de Vigilancia Tecnológica. En este país también existen empresas proveedoras de Vigilancia Tecnológica, entre ellas se encuentran IALE, CIEMAT, Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas, CIS Galicia e Infocenter (Angelozzi, y otros, 2011).

A finales del pasado siglo, Francia incursiona en la Vigilancia Tecnológica. En este país el proceso se caracteriza por su enfoque hacia la búsqueda de información que se pueda extraer de la competencia, haciendo uso de técnicas estadísticas para análisis de bases de datos y análisis semántico de textos; además en el desarrollo de software especializado para el análisis y representación de grandes volúmenes de información. Estados Unidos fue pionero en estos estudios, sus antecedentes datan de los años 60 por la necesidad de monitoreo del ambiente; en 1986 se crea la SCIP para el desarrollo profesional de la Inteligencia Competitiva o Vigilancia Competitiva y actualmente enfocada en desarrollar la competitividad a partir de la explotación de información científica y técnica (Lara Pérez, y otros, 2009).

En el mundo existen varias empresas líderes en materia de Vigilancia Tecnológica entre estas se encuentran: Motorola, Nokia, L'Oreal, Grupo REPSOL, Ferrer Internacional: Centro de Investigación Farmacéutica Grupo Ferrer (Martinez Negrín, 2012).

En el caso de Cuba, la Vigilancia Tecnológica se practica desde principios del siglo XX. El archivo histórico del Ministerio de Comercio Exterior atesora un ejemplar de un estudio de mercado realizado en 1901 en nuestro país. Según estudios realizados por los especialistas de la Dirección de Información, se evidencia que existen instituciones con gran desarrollo en la actividad de Vigilancia Tecnológica, entre las que se pueden mencionar:

Consultoría BioMundi (IDICT/CITMA), Disaic (SIME), Consultorías Delfos y Avante (MIC), varias universidades (Instituto Superior Politécnico José Antonio Echeverría, Universidad de la Habana y Universidad Central de las Villas), Instituto Finlay, Centro Cubano de Investigaciones Científicas (CNIC), Centro de Ingeniería Genética y Biotecnología (CIGB), Unión CUPET, Oficina de Información del Comité Central, CIGET de Villa Clara, Centro de Investigación y Desarrollo Técnico del MININT, CIGET de Granma e Instituto de Investigaciones de la Industria Alimentaria (Martínez Negrín, 2012).

Luego dada la inserción de las tecnologías en la gran mayoría de las esferas de gestión existen un conjunto de sistemas informáticos que van dirigidos a apoyar estos sistemas de VT como puede notarse a continuación.

1.3.1 Ejemplos de sistemas informáticos para Vigilancia Tecnológica a nivel internacional y nacional

Tanto a nivel nacional como internacional existen múltiples aplicaciones con el propósito de gestionar la VT en las empresas, muchas de estas son sitios como portales o intranets o la sinergia de estos.

Plataforma de Vigilancia Tecnológica: VIGIALE

VIGIALE es una plataforma que gestiona los servicios de la Vigilancia Tecnológica y constituye un punto de acceso a documentos, noticias, eventos, proyectos e información de interés general, ofreciendo además servicios de monitoreo activo de cambios en fuentes on-line, alertando sobre nuevas noticias, cambios en normativas, el mercado, y patentes a través de múltiples canales de notificación. Permite además gestionar pedidos personalizados de información, facilitando la comunicación entre los analistas del equipo de vigilancia y los usuarios.

Cuenta con una interfaz web simple, a través de la cual los usuarios en cualquiera de los roles puede acceder a todos los servicios de la vigilancia. Contempla la gestión de una amplia comunidad de usuarios, permitiendo la personalización de la información que se recibe e integra conceptos adelantados de redes sociales que van desde espacios para la discusión de temas de interés hasta la posibilidad de que los usuarios participen en la clasificación y la valoración de los informes mediante el uso de etiquetas dinámicas.

MADRI+D

El Sistema Madri+d es una red de trabajo que agrupa a instituciones públicas y privadas de investigación y a las asociaciones empresariales regionales, que cubre los aspectos esenciales de comunicación entre el sector productor de conocimiento y el sector industrial, con el objetivo de mejorar la competitividad de la región mediante la transferencia de conocimiento. Su puesta en marcha fue en 1997, desde allá a la fecha ha ido incorporando secciones y servicios. En su esfera, es líder en Iberoamérica, se puede acceder en inglés, portugués y español.

Los servicios de Información Tecnológica que ofrecen los Círculos de Innovación Madri+d son los siguientes (Mi+d, 2014):

- Informes de Vigilancia Tecnológica de diferente naturaleza en función del destinatario: estudios sectoriales que recogen revisiones de tecnologías en una determinada área cuyo objetivo es sensibilizar y movilizar a la industria para su implementación y estudios tecnológicos muy específicos dirigidos a NEBTs (nuevas empresas de base tecnológica) y/o grupos de investigación.
- Publicaciones divulgativas, editadas en colaboración con asociaciones empresariales.
- Servicios periódicos de información y alertas sobre noticias tecnológicas, nuevos productos y tecnologías, novedades legislativas, ayudas y subvenciones, proyectos en desarrollo.
- Creación y mantenimiento de bibliotecas on-line, donde se clasifican y actualizan: novedades tecnológicas, documentos clave y prospectivos, ofertas y demandas tecnológicas, eventos y boletines.

Observatorio Tecnológico del Instituto Tecnológico de Informática (ITI)¹

El Instituto Tecnológico de Informática (ITI) ha desarrollado el Observatorio Tecnológico del sector TIC a través de una infraestructura específica que permite recopilar, almacenar, categorizar y mostrar información científico-técnica relevante, que brinda a sus usuarios, ya sea una empresa o institución científica, la posibilidad de trazar planes y estrategias tecnológicas, minimizando la incertidumbre y potenciando de esta forma la innovación y el desarrollo. Entre las características más importantes de este Observatorio Tecnológico se encuentran (Saéz Domingo, y otros, 2008):

- **Portal Web del Observatorio:** Por defecto, en portada se muestran los cuatro tipos de recursos que tienen mayor caducidad temporal. Además, el portal ofrece la posibilidad de mostrar la información para una categoría concreta, para un tipo de recurso concreto o mediante el cruce de ambos (categoría-tipo). La información que se muestre en el portal dependerá de los permisos del usuario final.
- **Buscador:** Permite la búsqueda por texto libre en título, contenido y etiquetas asignadas a cada documento. Los resultados de la búsqueda se presentan agrupados por tipo de recurso.

¹ Disponible en <https://observatorio.iti.upv.es/>

- **Sindicación de contenidos:** Se han implementado mecanismos de sindicación RSS al contenido del Observatorio, contemplando la sindicación global (a todos los recursos), la sindicación a un tipo de recurso concreto y la sindicación a una categoría específica.
- **Envío de boletines electrónicos:** Previo registro y a través del correo electrónico, se puede recibir periódicamente la información más reciente. Los boletines podrán ser de tipo genérico, con toda la información de todas las categorías, o el propio usuario podrá personalizarlos indicando las categorías y el tipo de recurso de su interés.

Sistema de Vigilancia Tecnológica del Ministerio de la Informática y las Comunicaciones (SVT-MIC)

El SVT-MIC ha sido estructurado como un sistema integrado coordinado desde DELFOS, que agrupa a las organizaciones del MIC, que con funciones específicas deben generar de forma conjunta la información necesaria y pertinente, que pueda significar una oportunidad o amenaza para nuestro país en función de las Tecnologías de la Información y las Comunicaciones y que impliquen decisiones estratégicas (Font Rodríguez, y otros, 2008).

Este sistema se inserta en el Viceministerio que atiende las actividades de Investigación, Desarrollo e Innovación de Cuba, apoyando de esta forma el Sistema de Ciencia e Innovación Tecnológica y brindar además productos y servicios de Vigilancia Tecnológica a todos los decisores del Ministerio. El SVT-MIC está compuesto por un gran equipo de trabajo dedicado a garantizar el acceso a fuentes de información especializada y actualizada importante para los procesos de toma de decisiones, informes y alertas de la Vigilancia Tecnológica, aplicaciones y tutoriales de interés para el sector, y un grupo de informaciones relacionadas a eventos disponibles tanto en modalidad presencial como virtual.

BioMundi²

La Consultoría BioMundi es un centro especializado en brindar servicios y productos de información e inteligencia empresarial, que son de utilidad práctica en las esferas científica, tecnológica, económica y comercial. Para ello conjuga los modernos principios de la gestión de información y del conocimiento, y los desarrollos más actuales de las tecnologías de información. Apoya a las organizaciones para:

² Disponible en <http://biomundi.idict.cu>

- Mejorar la competitividad y la rentabilidad aprovechando las herramientas de Inteligencia Empresarial.
- Tomar decisiones en cuanto a sus estrategias sectoriales, corporativas, de negocios y de cambio.
- Capitalizar sobre las atractivas oportunidades ante el mercado y la competencia.
- Complementar la creatividad e innovación en la empresa.

La Consultoría BioMundi es una división del Instituto de Información Científica y Tecnológica (IDICT), del Ministerio de Ciencia, Tecnología y Medio Ambiente, de la República de Cuba, creada en 1992 como centro de servicios de inteligencia empresarial, para las organizaciones del Polo Científico del Oeste de La Habana, el cual constituye el clúster de la biotecnología cubana, así como para otros sectores de la economía.

Vigitec: Servicio de Vigilancia Tecnológica de la UCI

El Grupo de Vigilancia Tecnológica de la Dirección de Información de la Universidad de las Ciencias Informáticas, pone a disposición de toda la comunidad, numerosos productos y servicios de Vigilancia Tecnológica e Inteligencia Empresarial, cuya finalidad fundamental es garantizar una toma de decisiones documentada. Vigitec es la plataforma encargada de gestionar todos estos productos y servicios. Dichos productos y servicios están diseñados para apoyar todos los procesos desarrollados en la UCI: Docente, Productivo e Investigativo. Entre los servicios con los que cuenta están:

- **Servicio de alertas tecnológicas:** Parte de la observación, captación, análisis y diseminación sistemática de las noticias y artículos recuperados en bases de datos y en la Web. Compendios informativos: Se encarga de recopilar la información detallada acerca de determinada materia o temática, con un nivel de análisis exhaustivo. Puede incluir información gráfica y demás técnicas de visualización de la información.
- **Perfil de sector:** Contiene información sobre parámetros establecidos con el cliente acerca de las empresas de determinado sector. Sus vínculos comerciales, líneas de investigación y producción fundamentales, posición en el mercado, liderazgo y características principales de cada compañía del sector.
- **Investigaciones de mercado:** Recopilación, procesamiento y análisis de la información relativa a determinado mercado, que una vez interpretada reduce considerablemente los riesgos presentes en la toma de decisiones.
- **Análisis de tendencias:** Recopilación y análisis de la información histórica disponible en BD y otras fuentes. Selección de las variables a estudiar según los intereses del cliente. Tienen como objetivo determinar el estado actual y el comportamiento de determinado sector de

interés, relativos a la Investigación-Desarrollo, la Innovación Tecnológica y la comercialización.

Para un mejor análisis se tabula una matriz de identificación funcional, con el objetivo de identificar aquellas funcionalidades que estarán presentes en la solución que se propone, por un lado se muestran las aplicaciones descritas anteriormente y por otro los servicios a los que dan respuesta, como puede notarse en la Tabla 1.

Tabla 1: Comparación de los Sistemas Analizados

Servicios de Información	VIGIALE	Observatorio Tecnológico del ITI	Sistema de VT del MIC	Vigitec
Búsqueda de Información	Sí	Sí	Sí	Sí
Sindicación de Contenido (RSS)	Sí	Sí	No	Sí
Boletines Electrónicos	No	Sí	Sí	Sí
Publicación de Noticias	Sí	Sí	Sí	Sí
Catálogo de Tesis de Grado	No	No	No	No
Opinión de Expertos	No	No	Sí	Sí

Tras el análisis de la Tabla 1 puede verse cómo la mayoría de las aplicaciones analizadas prestan la mayor parte de los servicios de información que se muestran en la tabla; sin embargo ninguna de ellas cubre la totalidad de estos. Puede inferirse que es posible, y necesario, su implementación en una misma aplicación dado que son cubiertos por el 66% de las aplicaciones. Entonces se llega a la conclusión de que una novedad en la propuesta de solución radica en la integración, en una misma herramienta informática, de la totalidad de los servicios objeto de análisis en esta tabla.

Por las particularidades del problema a tratar y las características de la solución que se propone los autores determinaron hacer uso de las bondades de los gestores de contenido en el desarrollo de la misma.

1.4 Sistemas de Gestión de Contenidos (CMS)

1.4.1 Wordpress

Wordpress fue en sus inicios un CMS orientado al desarrollo de Blogs, pero ha evolucionado hacia uno de los CMS de uso general más completos del mercado. Su origen se encuentra en la herramienta b2/cafelog, diseñada por Michel Valdrighi, y pensada como un sistema para la gestión de blogs personales. La primera versión de Wordpress, la 0.7, surge en 2003, encontrándose actualmente en la versión 3.3.1. El modelo de actualizaciones ha sido hasta la fecha bastante dinámico con 3 o 4 versiones base anuales, y versiones menores con una frecuencia de actualización de aproximadamente un mes (Colectivo de Autores, 2012).

Es una herramienta muy interesante para crear y gestionar un blog en el espacio web de un centro o proyecto educativo. Permite la publicación de contenidos de una forma fácil. Para su instalación es necesario disponer de PHP y MySQL en el servidor. Las características más destacadas de Wordpress son (Dominguez, 2008):

- **Aplicación de Libre Distribución (GPL):** Se puede utilizar y modificar de forma gratuita.
- **Gestión de usuarios:** Permite crear usuarios y asignarles diferentes niveles de permisos en relación con la creación y publicación de contenidos en el blog.
- **Facilidad de instalación y actualización.**
- **Suscripción RSS:** Soporta el sistema de noticias RSS de tal forma que cualquier lector de noticias RSS puede suscribirse a los artículos, comentarios.
- **Personalización del diseño:** Se pueden utilizar plantillas y temas editables que permiten personalizar el aspecto gráfico del blog.
- **Creación de contenidos:** Admite proteger la lectura de un artículo con una contraseña; crear un artículo y que se publique al llegar una fecha; segmentar en partes un artículo demasiado largo para evitar un scroll vertical excesivo; vista previa de artículos sin necesidad de publicarlo; subir e insertar imágenes propias y miniaturas; subir y enlazar distintos tipos de archivos (pdf, zip, doc); organizar los artículos en categorías o secciones; gestionar borradores de artículos que se pueden guardar para retomarlos más adelante y publicarlos cuando se estime conveniente.
- **Favoritos:** Al pulsar en el marcador de página asociado a cada artículo se crea en el navegador un enlace directo a esa página.
- **Archivo histórico:** Muestra de forma organizada en el tiempo los artículos anteriores.

- **Búsqueda:** Integra una herramienta de búsqueda que permite a los visitantes buscar en todo el blog por un término que se introduce como dato.
- **Comentarios:** Los visitantes pueden introducir sus comentarios a un artículo. Estos comentarios admiten algunas etiquetas HTML para enriquecer su formato de presentación. También admiten moderación y ofrecen el servicio de notificación por email a los usuarios suscritos cada vez que se introduzca uno nuevo.

El amplio uso de este CMS se refleja en los siguiente sitios profesionales que reciben cientos de miles de visitas mensuales:

- CNN Politics: <http://cnnespanol.cnn.com/>
- La revista Time: <http://thepage.time.com/>
- La revista People: <http://stylenews.peoplestylewatch.com/>
- El Blog de Flickr: <http://blog.flickr.net/es>

Otra de las características más importantes de este CMS es la protección de la privacidad de los contenidos, a través de la definición de niveles de usuario, protección de contenidos por contraseña, filtros antispam o controles de comentarios. Además permite generar estadísticas de acceso al sitio Web: número de visitantes, lugar de origen de las visitas, páginas visitadas y tiempo de acceso. Por último destacar que Wordpress dispone de un buen soporte de ayuda a través de abundante documentación y foros.

1.4.2 Joomla

Joomla es un CMS premiado mundialmente, que permite la construcción de sitios web y otras aplicaciones online potentes. Es una solución de código abierto y está disponible libremente para cualquiera que desee utilizarlo. Resulta muy sencillo para usuarios sin conocimientos técnicos, ya que les permite a estos añadir o editar contenido, subir imágenes, y gestionar los datos principales de su compañía u organización (Joomla, 2014).

Este gestor de contenidos surge en 2005 como resultado de una división del proyecto Mambo. A partir de esta decisión, muchos colaboradores, comunidades y diseñadores, respaldaron el proyecto, qué evolucionó hasta convertirse en lo que es hoy en día, uno de los CMS mas usados y conocidos del mercado. Para extraer todo el potencial de Joomla, se requiere cierto conocimiento y experiencia, ya que su máxima versatilidad se obtiene de la integración, adaptación y desarrollo de nuevos módulos. Sin lugar a dudas uno de los puntos fuertes de Joomla es su magnífica comunidad. Fruto de la gran participación de los usuarios, el sistema se encuentra en continua actualización frente a vulnerabilidad, bugs, nuevas funcionalidades y extensiones. Gracias también a la fortaleza de su

comunidad se dispone de un soporte muy completo a través de Webs oficiales, foros, y todo tipo de documentación generada (Colectivo de Autores, 2012).

Joomla es utilizado para el desarrollo de sitios web de diferentes tipos, como (Joomla, 2014):

- Sitios Web Corporativos o Portales
- Comercio Electrónico
- Pequeños sitios de egocios
- Aplicaciones Gubernamentales
- Intranets Corporativas
- Sitios web para escuelas
- Páginas personales o familiares
- Portales de Comunidades
- Revistas y Periódicos

Entre sus principales características se encuentran las siguientes (Graf, y otros, 2012):

- Componente Finder para realizar búsquedas
- Drivers para MySQL y Microsoft SQL Server
- Opciones para el Modo Offline
- Registro de usuarios con CAPTCHA
- Manejo para diseños simples
- Notas de usuario
- Optimización para motores de búsqueda

Con respecto a las características técnicas, cabe mencionar que Joomla se encuentra liberado bajo una licencia GPL y utiliza PHP como lenguaje de programación, MySQL como motor de base de datos y Apache como servidor Web.

1.4.3 Drupal

El CMS Drupal es un sistema de gestión de contenidos web muy versátil, con licencia GNU/GPL, escrito en PHP, desarrollado y mantenido por una activa comunidad de usuarios. Destaca por su flexibilidad, adaptabilidad y por su interfaz de administración que permite crear fácilmente nuevos tipos de contenido estructurados, así como la gran cantidad de módulos adicionales disponibles, hace que sea adecuado para realizar diferentes tipos de sitios web, tiendas online y foros. Su desarrollo fue

iniciado por Dries Buytaert en 1999 y no fue hasta 2001 cuando se publicó la primera versión del CMS. Actualmente Drupal se encuentra en la versión 7.24. Entre las principales páginas web desarrolladas con Drupal están (Dlógica, 2013):

- La casa blanca: <http://www.whitehouse.gov/>
- Ubuntu: <http://www.ubuntu.com/>
- Nesto: <http://www.nesto.dk/>
- Observer: <http://observer.com/>
- MTV: <http://www.mtv.co.uk/>

El CMS Drupal cuenta con un sistema de ayuda online y páginas de ayuda para los módulos del núcleo, tanto para usuarios como para administradores. Todo el contenido en Drupal es totalmente indexado en tiempo real y se puede consultar en cualquier momento. Dispone de un entorno de personalización robusto, tanto el contenido como la presentación pueden ser tratados de forma individual de acuerdo a unas preferencias definidas por el usuario. La gestión de contenido se realiza como objetos independientes, de forma que puede realizarse un tratamiento individualizado de la información, facilitando su inclusión en cualquier página o permitiendo comentarios específicos sobre cada uno de ellos.

Permite actualizar el contenido de forma sencilla. Brinda la posibilidad de gestionar las taxonomías y estructurar los contenidos de forma personalizable, además cuenta con un sistema de caché avanzado.

Drupal usa el *mod_rewrite* de Apache para crear URLs que son manejables por los usuarios y los motores de búsqueda. Para su uso en una intranet, Drupal se puede integrar con un servidor LDAP. Los administradores de Drupal no tienen que establecer permisos para cada usuario. En lugar de eso, pueden asignar permisos a un rol determinado y agrupar los usuarios por roles. A continuación se muestran otras de las características que posee este CMS (Hispano, 2013):

- Control de versiones
- Enlaces permanentes (*Percalinas*)
- Objetos de Contenido (Nodos)
- Plantillas (*Templates*)
- Sindicación del contenido
- Soporte de *Blogger*
- Independencia de la base de datos
- Análisis, seguimiento y estadísticas

Selección del CMS a utilizar para el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL

Partiendo del análisis realizado anteriormente acerca de las herramientas que permiten el trabajo con Sistemas de Vigilancia Tecnológica, se selecciona para el desarrollo de la solución el CMS Drupal. Este permite la creación, desarrollo y mantenimiento de servicios y productos de información digital; además brinda la posibilidad de publicar contenidos por RSS. En cuanto a la administración de roles y usuarios, brinda la posibilidad de asignar permisos y agrupar los usuarios por roles. También permite crear estructuras para la recuperación de información y tiene un sistema de taxonomías fuerte que ofrece la posibilidad de categorizar contenidos. Es un sistema flexible a la hora de integrarse con diferentes servidores Web y diferentes sistemas de Bases de Datos.

1.5 Metodologías de Desarrollo de Software

Una metodología de desarrollo, en ingeniería de software, es un conjunto de herramientas, técnicas, procedimientos y soporte documental encaminados a estructurar, planificar y controlar el proceso de desarrollo de forma organizada y lógica, que tienen como objetivo apoyar a los desarrolladores en la creación de un nuevo software (Kaisler, 2005).

Las metodologías de desarrollo se clasifican en dos clases: las metodologías tradicionales o robustas y las ágiles o ligeras.

Metodologías tradicionales

Las metodologías tradicionales o prescriptivas definen un conjunto de actividades, acciones, tareas, fundamentos y productos de trabajo que se requieren para desarrollar software de alta calidad. El proceso conduce a un equipo de software a través de un conjunto de actividades del marco de trabajo que se organizan en un flujo de proceso, el cual puede ser lineal, incremental o evolutivo. La terminología y los detalles de cada modelo difieren, pero las actividades genéricas del marco de trabajo permanecen razonablemente consistentes (Pressman, 2005).

Metodologías ágiles

Las metodologías ágiles combinan una filosofía y conjunto de directrices de desarrollo. La filosofía busca la satisfacción del cliente y la entrega temprana de software incremental, equipos de proyecto pequeños y con alta motivación, métodos informales, un mínimo de productos de trabajo de la ingeniería de software, y una simplicidad general de desarrollo. Las directrices de desarrollo resaltan la entrega sobre análisis y diseño (aunque estas actividades no se descartan) y la comunicación activa entre los desarrolladores y los clientes (Pressman, 2005).

La Tabla 2 muestra una comparación entre las metodologías ágiles y tradicionales (Letelier, y otros, 2012).

Tabla 2: Comparación entre Metodologías Tradicionales y Ágiles

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de software	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparado para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo de desarrollo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas y normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (menos de 10 estudiantes) trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura de software	La arquitectura de software es esencial y se expresa mediante modelos

1.5.1 Extreme Programming (XP)

XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (Letelier, y otros, 2012).

Uno de los propósitos de XP es reducir los costos ante los cambios que puedan surgir durante la vida de un proyecto. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y la aplicación de las siguientes prácticas:

- 1) El juego de la planificación:** el alcance de la siguiente versión está definido por consideraciones del negocio (prioridad de los módulos, fechas de entrega) y estimaciones técnicas (estimaciones de funciones, consecuencias). El objetivo del juego es maximizar el valor del software producido.
- 2) Versiones pequeñas:** un sistema simple se pone rápidamente en producción. Periódicamente se producen nuevas versiones agregando en cada iteración aquellas funciones consideradas valiosas para el cliente.
- 3) Metáfora del sistema:** cada proyecto es guiado por una historia simple que explica el funcionamiento del sistema en general, reemplaza a la arquitectura y debe estar en lenguaje común, entendible para todos, cliente y desarrolladores.
- 4) Diseño simple:** el sistema se diseña con la máxima simplicidad posible. Se plasma el diseño en las tarjetas CRC (Clase - Responsabilidad - Colaboración), con lo que las clases definidas durante el análisis pueden ser filtradas para determinar las que son realmente necesarias para el sistema.
- 5) Pruebas continuas:** los casos de prueba se escriben antes que el código. Los desarrolladores escriben pruebas unitarias y los clientes especifican pruebas funcionales.
- 6) Refactorización:** es posible reestructurar el sistema sin cambiar su comportamiento, por ejemplo eliminando código duplicado, simplificando funciones.
- 7) Programación por parejas:** el código es escrito por dos personas trabajando en la misma computadora.
- 8) Posesión colectiva del código:** nadie es dueño de un módulo, cualquier programador puede cambiar cualquier parte del sistema en cualquier momento, por ello siempre se deben utilizar estándares de codificación.
- 9) Integración continua:** los cambios se integran en el código base varias veces por día. Todos los casos de prueba deben ejecutarse antes y después de la integración. Se dispone de una máquina para la integración y se realizan pruebas funcionales donde participa el cliente.
- 10) Cliente en el sitio:** el equipo de desarrollo tiene acceso todo el tiempo al cliente, el cual está disponible para responder preguntas y fijar prioridades.
- 11) Estándares de codificación:** todo el código debe estar escrito de acuerdo a un estándar de codificación.

XP define *UserStories* (Historias de Usuarios) como base del software a desarrollar. Estas historias las escribe el cliente y describen escenarios sobre el funcionamiento del software (Díaz Flores, 2013). Existen diferentes roles (actores) y responsabilidades para diferentes tareas y propósitos definidos en la metodología XP de acuerdo con la propuesta original de Kent Beck. Ejemplos de ello son:

Programador (*Programmer*)

- Responsable de decisiones técnicas.
- Responsable de construir el sistema.
- Sin distinción entre analistas, diseñadores o codificadores.
- En XP, los programadores diseñan, programan y realizan las pruebas.

Cliente (*Customer*)

- Es parte del equipo.
- Determina qué construir y cuándo.
- Escribe tests funcionales para determinar cuándo está completo un determinado aspecto.

Entrenador (*Coach*)

- El líder del equipo, toma las decisiones importantes.
- Principal responsable del proceso.
- Tiende a estar en un segundo plano a medida que el equipo madura.

Rastreador (*Tracker*)

- Observa sin molestar.
- Conserva datos históricos.

Probador (*Tester*)

- Ayuda al cliente con las pruebas funcionales.
- Se asegura de que los *tests* funcionales se ejecutan.

1.5.2 Rational Unified Process (RUP) Ultra-Light

RUP es un proceso muy organizado; pero que requiere mucha documentación y para emplearlo es necesario adaptarlo a las necesidades de la entidad que lo utilice. Con el fin de agilizar este proceso, muchas personas han tomado los elementos más significativos del mismo y es cuando empieza a surgir el término de *RUP Ultra Light*, esta metodología no es más que una adaptación del Proceso

Unificado de Desarrollo de Software (RUP), integrada por 10 pasos enmarcados en 4 etapas: análisis, diseño, programación y puesta en marcha.

Se deben seguir los siguientes 10 pasos para desarrollar la metodología de desarrollo de software *RUP Ultra Light* (García Pérez, y otros, 2010):

1. Realizar un Diagrama de Casos de Uso.
2. Priorizar los Casos de Uso a trabajar.
3. Generar los Documentos de Caso de Uso.
4. Generar los Diagramas de Secuencia.
5. Diseñar el Framework del proyecto.
6. Creación de la Base de Datos.
7. Construir la Aplicación Desktop.
8. Programar las funcionalidades de los Casos de Uso.
9. Probar los Requerimientos del Software.
10. Integrar los requerimientos concluidos.

Se deben repetir de los pasos 3 al 10 por cada iteración que se haya programado para el proyecto, para de esta forma poder controlarlo.

1.5.3 Scrum

Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos que se ha utilizado con éxito durante los últimos 10 años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. Otra característica importante son las reuniones a lo largo del proyecto. Estas son las verdaderas protagonistas, especialmente la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración (Kniberg, 2007).

Scrum es una metodología ágil, y como tal (Palacios, 2011):

- Es un modo de desarrollo de carácter adaptable más que predictivo.
- Orientado a las personas más que a los procesos.
- Emplea la estructura de desarrollo ágil: incremental basada en iteraciones y revisiones.

Selección de la metodología de desarrollo de software para el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL

Para el proceso de desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL se seleccionó la metodología de desarrollo de software Extreme Programming (XP) debido su utilidad para proyectos de corto plazo y corto equipo. La metodología consiste en una programación rápida o extrema donde el cliente forma parte del equipo de desarrollo, uno de los requisitos para alcanzar el éxito del proyecto y la satisfacción del cliente. Además tiene como ventajas la simplicidad, la comunicación entre los usuarios y los desarrolladores y la retroalimentación concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales. Todo el proceso de esta metodología está encaminado a conseguir la meta final, el software, otorgándole gran importancia a las relaciones interpersonales de los miembros del equipo de desarrollo, así como a la velocidad de reacción ante los constantes cambios que puedan surgir durante el desarrollo.

1.6 Lenguajes para el Desarrollo

HyperText Markup Language (HTML)

HTML es un lenguaje de programación que se utiliza para el desarrollo de páginas de Internet. Se trata de la sigla que corresponde a *HyperText Markup Language*, es decir, Lenguaje de Marcas de Hipertexto, que podría ser traducido como Lenguaje de Formato de Documentos para Hipertexto. El lenguaje *HTML* es un estándar reconocido en todo el mundo y sus normas están definidas por el *World Wide Web Consortium* que es un organismo sin ánimo de lucro conocido mundialmente como *W3C*. Este lenguaje se utiliza en muchas aplicaciones electrónicas como buscadores, tiendas online y banca electrónica (Eguiluz, 2014).

Hojas de Estilo en Cascada (CSS)

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

CSS se utiliza para dar estilo a documentos *HTML* y *XML*, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos *HTML* y *XML*. CSS permite a los desarrolladores Web controlar el estilo y el formato de múltiples páginas Web al mismo tiempo (W3C, 2008).

JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos cómo: texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de

aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Pérez, 2009).

PHP

El *PHP (Hypertext Preprocessor)*, es un lenguaje interpretado de alto nivel embebido en páginas *HTML* y ejecutado en el servidor. El *PHP* se inició como una modificación a *Perl* escrita por Rasmus Lerdorf a finales de 1994. Una de sus características más potentes es su soporte para gran cantidad de bases de datos, entre ellas *InterBase*, *mSQL*, *MySQL*, *Oracle*, *Informix*, *PostgreSQL*. *PHP* también ofrece la integración con varias bibliotecas externas, que permiten que el desarrollador haga casi cualquier cosa, desde generar documentos en PDF hasta analizar código *XML*. Es un producto de código abierto y es utilizado en aplicaciones web relacionadas por algunas de las organizaciones más prominentes tales como *Mitsubishi*, *Redhat*, *Der Spiegel*, *MP3-Lycos*, *Ericsson* y *NASA*³ (Henst, 2001).

1.7 Herramientas CASE

Las herramientas *CASE (Computer Aided Software Engineering)* son un conjunto de programas informáticos que ayudan a los analistas, ingenieros de software y desarrolladores durante todos los pasos del ciclo de vida del desarrollo de un proyecto, desde la planificación pasando por el análisis y el diseño hasta la generación del código fuente de los programas y la documentación.

Dentro de las herramientas *CASE* que sirven para modelar el sistema se encuentran los Lenguajes Unificados de Modelado (*UML*) que son un lenguaje para el desarrollo de software orientado a objetos, su propósito es visualizar, especificar, construir y documentar proyectos de software. Entre las herramientas *CASE* orientadas a *UML* se encuentran:

1.7.1 Visual Paradigm

Es una herramienta *UML* profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Tiene como ventaja que permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales, demostraciones

³ *National Aeronautics and Space Administration* (Administración Nacional de la Aeronáutica y del Espacio)

interactivas y proyectos *UML*. Presenta licencia gratuita y comercial. Es fácil de instalar, actualizar y es compatible entre ediciones.

1.7.2 ArgoUML

Es una herramienta de código abierto utilizada para el modelado *UML*, es muy sencilla y se puede utilizar para aplicar ingeniería inversa. Exporta diagramas a diferentes formatos de imágenes y tiene soporte para *OCL*⁴. Cuenta con soporte para bases de datos. Está publicada bajo la licencia *BSD* y la plataforma que soporta es Java. Proporciona la generación del código que puede ser empleado en los siguientes lenguajes: *Java*, *C++*, *C Sharp* y *PHP*.

1.8 Entorno de Desarrollo Integrado (IDE)

1.8.1 Netbeans IDE

Un entorno de desarrollo integrado, también llamado *IDE* por su sigla en inglés (*Integrated Development Environment*), es un programa informático compuesto por un conjunto de herramientas de programación como: los compiladores, editores y depuradores; que están disponibles para prestar su apoyo en la mayoría de los lenguajes de programación convencionales. Entre estos se encuentra el *NetBeans Platform* con una amplia variedad de APIs que resuelven gran cantidad de problemas que se presentan al construir una aplicación.

Es un *IDE* multiplataforma, de código abierto, pensado para escribir, compilar, depurar y ejecutar software. Sus funciones están provistas por módulos. Está escrito y pensado para Java; pero brinda soporte para los lenguajes de programación: *C/C++*, *Ruby*, *Python*, *HTML*, *CSS* y *PHP* (Guervos Merelo, 2005).

1.8.2 Zend Studio

Es un *IDE* escrito en el lenguaje de programación Java, multiplataforma y diseñado para maximizar la productividad de los desarrolladores. Permite la gestión de proyectos e integra el Sistema de Control de Versiones (*CVS*), y tiene integrado un cliente *FTP*. Brinda soporte para *PHP 4* y *PHP 5*. Contiene una ayuda contextual con todas las librerías de funciones, permitiendo el auto completamiento de código y el resaltado de sintaxis. Detecta errores sintácticos en tiempo real y posee funciones de depuración que muestran el contenido de variables, salidas de “*script*”, alertas y errores.

⁴ Lenguaje para la descripción formal de expresiones en los modelos UML.

Selección de las herramientas de desarrollo para el Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL

Durante el estudio realizado se decidió utilizar la herramienta *CASE Visual Paradigm* en su versión 5.0 para la modelación visual del sistema ya que es un producto de calidad que soporta aplicaciones web, genera código desde diagramas y genera documentación. Es una herramienta que soporta gráficos en varios formatos, contiene varios idiomas y permite la exportación como *HTML*. Es fácil de instalar y actualizar, además es personalizable. Para la programación del sistema se decidió utilizar el *IDE Netbeans* en su versión 7.4 por sus probadas características para el desarrollo de software, multiplataforma, de código abierto y con soporte para *HTML, CSS* y *PHP*. Además realiza completamiento de código lo que facilita el trabajo de los programadores.

1.9 Servidor Web

Apache

Es un servidor web de código abierto, altamente configurable y modular. Utiliza *Perl, PHP* y otros lenguajes. Soporta varios sistemas operativos: *Linux, Solaris, Mac OS* y *Windows*. Su función principal es analizar cualquier archivo solicitado por un navegador y mostrar resultados correctos de acuerdo con el código del archivo. Permite configurar los informes de errores, presenta visualización de códigos en numerosos niveles de *HTML* y la capacidad de determinar qué nivel del navegador puede aceptar el contenido. Es compatible con *CGI*⁵, permite la personalización de variables y tiene soporte para reparación de errores. Es uno de los primeros servidores en soportar host basados en direcciones IP y host virtuales. Tiene un elaborado índice de directorios, un directorio de alias, comprobación de ortografía de las *URL* y manuales en la web (Naramore, 2005).

1.10 Sistemas Gestores de Bases de Datos (SGBD)

1.10.1 PostgreSQL

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacional (ORDBMS) basado en el proyecto POSTGRES, de la universidad de Berkeley. El director de este proyecto es el profesor Michael Stonebraker, y fue patrocinado por *Defense Advanced Research Projects Agency (DARPA)*, el *Army Research Office (ARO)* y el *National Science Foundation (NSF)*. Es una derivación libre (*OpenSource*) de este proyecto, y utiliza el lenguaje *SQL92/SQL99* (Pecos, 2010).

⁵ Interfaz de entrada común (en inglés *Common Gateway Interface*).

Entre sus principales características se encuentran (Blog, 2012):

- Es una base de datos 100% ACID (Atomicidad, Coherencia, Aislamiento, Durabilidad).
- Soporta distintos tipos de datos.
- Incluye herencia entre tablas.
- Copias de seguridad en caliente (*Online/hot backups*).
- Juegos de caracteres internacionales.
- *Multi-Version Concurrency Control (MVCC)*.
- Múltiples métodos de autenticación.
- Acceso encriptado vía *SSL*.
- Completa documentación.
- Licencia *BSD*.
- Multiplataforma.

Otra de las características más relevantes de este gestor es que utiliza multiprocesos lo que permite garantizar que el sistema sea estable, además de ser fiable y contar con integridad en los datos.

1.10.2 MySQL

MySQL es el sistema de gestión de bases de datos SQL (Lenguaje de Consulta Estructurado) *Open Source* más popular, lo desarrolla, distribuye y soporta *MySQL AB*. *MySQL AB* es una compañía comercial, fundada por los desarrolladores de *MySQL*. Es una compañía *Open Source* de segunda generación que une los valores y metodología *Open Source* con un exitoso modelo de negocio. También puede integrarse con varios lenguajes contando con interfaces de programación de aplicaciones (*APIs*) disponibles para *C*, *C++*, *Eiffel*, *Java*, *Perl*, *PHP*, *Python*, *Ruby*, y *Tcl*. Cuenta con un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite verificación basada en el host. Brinda soporte a grandes bases de datos. Además cuenta con registros de longitud fija y longitud variable (*MySQL*, 2011).

Selección del Sistema Gestor de Bases de Datos (SGBD) para el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL

Teniendo en cuenta el análisis de los SGBD *MySQL* y *PostgreSQL* para el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL se decide seleccionar *PostgreSQL* en su versión 9.2.4 ya que trae avances en materia de seguridad, cuenta con una documentación completa, soporte de aplicaciones, presenta una buena integridad de los datos evitando de esta forma pérdidas de datos en la aplicación, además permite la supervisión, mejor rendimiento y almacenamiento de datos especializados. Es multiplataforma y debido a su escalabilidad puede mantener una velocidad estable.

1.11 Conclusiones Parciales

Teniendo en cuenta lo estudiado acerca de los procesos de la Vigilancia Tecnológica se concluye que los servicios más adecuados para la propuesta a implementar son: alertas informativas, compilaciones de noticias e informaciones científicas y técnicas, búsqueda de información especializada, obtención de artículos científicos y técnicos. Además se identificó la VT de Tecnologías como la clasificación más apropiada para la propuesta de solución teniendo en cuenta el objeto de vigilancia.

Las herramientas y tecnologías seleccionadas para el desarrollo de la solución fueron: el CMS Drupal en su versión 7.24, la metodología de desarrollo de software Extreme Programming (XP), los lenguajes de desarrollo PHP 5.3.8, HTML 5 y CSS 3, el Visual Paradigm 5.0 para el modelado, Netbeans 7.4 como entorno de desarrollo, Apache 2.2.21 como servidor web y PostgreSQL 9.2.4 como sistema gestor de bases de datos.

CAPÍTULO 2: PROPUESTA DEL SISTEMA DE APOYO A LA VIGILANCIA TECNOLÓGICA EN EL CENTRO DE GOBIERNO ELECTRÓNICO (CEGEL).

En el presente capítulo se realiza la propuesta del sistema. Para lograr dicho objetivo se describen las características del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL y los servicios que debe brindar, el modelo de dominio, las funcionalidades y las historias de usuario asociadas a cada funcionalidad. Se abordan además las dos primeras fases de la metodología de desarrollo empleada en la implementación de la aplicación que ha sido propuesta, denominadas exploración y planificación. También se muestran los artefactos que se generan durante la implementación de estas dos fases. Además se describen la arquitectura y los patrones de diseño que se van a utilizar en el desarrollo de la solución.

2.1 Descripción del Sistema

La propuesta del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL nace con el propósito de dotar a dicho centro de un sistema que permita tener disponible información actualizada relacionada con los productos que se desarrollan, de interés para los gestores de proyectos y su equipo de trabajo. Esta aplicación debe permitir gestionar la información actualizada y estratégica de aplicaciones para la toma de decisiones así como servir de soporte del proceso de Investigación, Producción e Innovación Tecnológica del CEGEL, fomentar el trabajo investigativo, realizar la búsqueda y consulta de las Tesis de Grado desarrolladas en el mismo. Se pretende lograr que todo el personal del centro se mantenga actualizado acerca del desarrollo científico-tecnológico internacional y nacional a través de noticias y otros servicios de información. Busca concentrar toda la información y garantizar el acceso a esta sin distinción, además de proveer una única interfaz de búsqueda y funcionalidades acordes con las características dinámicas de la web en estos tiempos.

Personas que interactúan con el sistema

Administrador: Tiene todos los privilegios del sistema, será el encargado de gestionar los roles correspondientes a cada usuario que se registre en la aplicación. También se encargará del mantenimiento de la base de datos.

Publicador: Este usuario gestiona todo el contenido concerniente al Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL, o sea Noticias y Tesis de Grado, además es el encargado de publicar los boletines tecnológicos del sitio. Una vez autenticado en el sistema podrá cambiar todo el contenido al que su nivel de privilegio le permita acceder.

Especialista de CEGEL: Este usuario posee privilegios mínimos, como: autenticarse en el sistema, emitir comentarios y consultar el contenido publicado.

Usuario Externo: Este usuario actúa como un usuario invitado al sistema, que navega por él sin autenticarse. Solo podrá hacer uso de los servicios que brinda el sitio en caso de que lo necesite.

Metáfora para el sistema

Los equipos de desarrollo en XP desarrollan una visión común sobre cómo funciona el programa, llamada "metáfora". La misma es una descripción evocativa simple sobre cómo funciona el programa; por ejemplo "este programa funciona como una colmena de abejas, que salen a buscar polen y lo traen de vuelta a la colmena", podría ser una descripción para un sistema de recuperación de información a través de agentes (Jeffries, 2010).

A veces no aparece una metáfora poética. En ese caso, con o sin una imagen viva, los equipos de desarrollo en XP usan un sistema común de nombres para asegurarse que todos entiendan cómo funciona el sistema, dónde buscar la funcionalidad que se desea, o cómo encontrar el lugar adecuado para agregar algo nuevo (Jeffries, 2010).

La metáfora seleccionada para el Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL fue: sistema que funciona como una industria obteniendo los abastecimientos de distintas fuentes, procesándolos y mostrando un producto final bien elaborado. En este caso los abastecimientos se refieren a las noticias extraídas directamente de los sitios de Ciencia y Tecnología y a las distintas Tesis de Grado desarrolladas en el CEGEL; y el producto final sería la forma de presentar los recursos de información.

Una vez desarrollada una visión general del funcionamiento del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL, se confeccionó el modelo de dominio para una mayor comprensión del mismo.

2.2 Modelo de Dominio

El modelo de dominio es utilizado como un medio para comprender el sector de negocios al que el sistema va a servir. El modelo de dominio puede ser tomado como el punto de partida para el diseño de la aplicación. Se basa en la representación de los conceptos más importantes y significativos para el desarrollo de un sistema. Este representa clases conceptuales del dominio del problema y conceptos del mundo real, no de los componentes de software. Además se definen las interrelaciones de los objetos más importantes representados mediante clases. A continuación se muestra el modelo de dominio del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL.

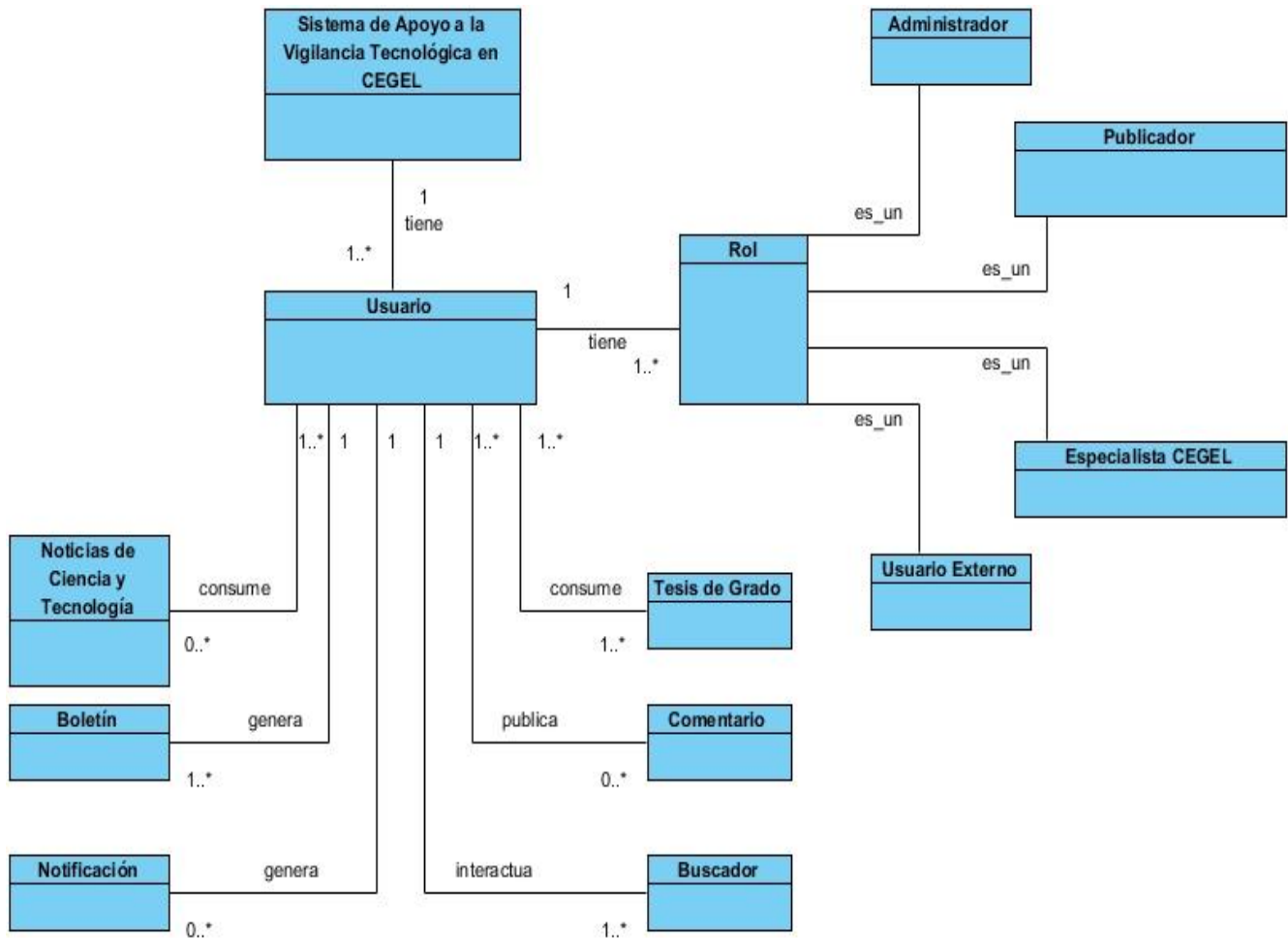


Figura 1: Modelo de Dominio

2.2.1 Descripción de las clases del modelo de dominio

Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL: Es el encargado de gestionar toda la información que se mostrará en el sitio, es decir, noticias, catálogo de tesis de grado y otros servicios de información.

Usuario: Son todas las personas que acceden al sistema. Estos serán tomados del servidor LDAP de la universidad para facilitar su gestión.

Rol: Identifican los niveles de acceso al sistema, estos permitirán a los usuarios navegar en él hasta ciertos niveles de acceso. Estos roles pueden ser: Administrador, Publicador, Especialista CEGEL y Usuario Externo.

Noticia: Es el relato o la redacción de un texto informativo que se refiere a un hecho novedoso, son informaciones provenientes del exterior del sistema.

Tesis de Grado: Son todos los documentos en formato digital de las Tesis de Grado realizadas en el CEGEL

Boletín: El boletín es un documento en formato pdf que recoge las noticias y los sucesos más relevantes del sistema.

Notificación: Son los avisos que genera el sistema para informarle al usuario del resultado inmediato de sus acciones sobre el sitio.

Comentarios: En los comentarios se recogen todas las opiniones, criterios y sugerencias que realizan todos los que interactúan con el sistema.

Buscador externo: Es un enlace a uno o varios buscadores externos, por ejemplo: Google, Bing. Esto será otra opción de búsqueda de información para los usuarios.

2.3 Especificación de Requisitos del Sistema

En la fase de análisis de requisitos se deben identificar claramente las necesidades del producto que se va a desarrollar. La especificación de requisitos es el resultado de esta fase en el que se describe lo que el futuro sistema debe hacer. El análisis de requisitos se puede definir como el proceso del estudio de las necesidades de los usuarios para llegar a una definición de los requisitos del sistema, hardware o software, así como el proceso de estudio y refinamiento de dichos requisitos. Los requisitos son una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado. La especificación de requisitos tiene como objetivos:

- Ayudar a los clientes a describir claramente lo que se desea obtener mediante un determinado software.
- Ayudar a los desarrolladores a entender qué quiere exactamente el cliente.
- Servir de base para desarrollos de estándares de especificación de requisitos particulares para cada organización.

Una buena especificación de requisitos permite la reducción del esfuerzo en el desarrollo, una buena base para la estimación de costes y planificación, un punto de referencia para procesos de verificación y validación, y una base para la identificación de posibles mejoras en los procesos analizados (Agut, 2001).

2.3.1 Requisitos Funcionales (RF)

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Se identificaron un total de 25 requisitos funcionales agrupados en 10 historias de usuario que se relacionan en la Tabla 2 que se muestra a continuación:

Tabla 3: Requisitos Funcionales

<p>HU 1: Autenticar usuario</p> <ul style="list-style-type: none"> ➤ RF 1: Autenticar usuario <p>HU 2: Gestionar usuario</p> <ul style="list-style-type: none"> ➤ RF 2: Adicionar usuario ➤ RF 3: Buscar usuario ➤ RF 4: Editar usuario ➤ RF 5: Eliminar usuario ➤ RF 6: Listar usuario <p>HU 3: Gestionar roles.</p> <ul style="list-style-type: none"> ➤ RF 7: Adicionar rol ➤ RF 8: Buscar rol ➤ RF 9: Eliminar rol ➤ RF 10: Editar rol <p>HU 4: Gestionar Tesis de Grado</p> <ul style="list-style-type: none"> ➤ RF 11: Adicionar Tesis de Grado ➤ RF 12: Buscar Tesis de Grado ➤ RF 13: Editar Tesis de Grado ➤ RF 14 Listar Tesis de Grado 	<p>HU 5: Gestionar comentarios</p> <ul style="list-style-type: none"> ➤ RF 15: Adicionar comentarios ➤ RF 16: Buscar comentarios ➤ RF 17: Eliminar comentarios ➤ RF 18: Editar comentarios <p>HU 6: Gestionar noticias</p> <ul style="list-style-type: none"> ➤ RF 19: Adicionar noticia ➤ RF 20: Editar noticia ➤ RF 21: Eliminar noticia <p>HU 7: Clasificar Tesis de Grado</p> <ul style="list-style-type: none"> ➤ RF 22: Clasificar Tesis de Grado <p>HU 8: Generar boletín</p> <ul style="list-style-type: none"> ➤ RF 23: Generar boletín <p>HU 9: Generar notificaciones</p> <ul style="list-style-type: none"> ➤ RF 24: Generar notificaciones <p>HU 10: Interactuar con buscadores genéricos</p> <ul style="list-style-type: none"> ➤ RF 25: Interactuar con buscadores genéricos
---	--

2.3.2 Requisitos No Funcionales (RNF)

Fiabilidad

- Se espera que el sistema sea tolerante a fallas.
- En caso de que el sistema presente alguna falla, los errores se deben mostrar sin detalles de información que pueda comprometer la seguridad e integridad del mismo.
- El tiempo medio de corrección de errores no debe exceder las 48 horas.
- Se espera un tiempo medio entre fallos de 35 días.

Eficiencia

- El sistema deberá tener una latencia, de cero (0) a cuatro (4) segundos aproximadamente.

Software

- Base de datos: PostgreSQL 9.2.4
- PHP Versión: 5.3.13 o superior
- PHP Memoria: 512 MB
- Servidor web Apache en el servidor

- Navegador web instalado en el cliente (para una mejor visualización se recomienda el navegador Mozilla Firefox)

Hardware mínimo para el cliente

- 512 MB de memoria RAM
- 20 GB de disco duro
- Procesador Pentium IV

Hardware mínimo para el servidor Web

- 6 GB de memoria RAM
- 60 GB de disco duro
- Intel Core 2 Duo o equivalente

Soporte

- Debe ser posible realizarle mantenimiento al sistema, así como incorporarle nuevas funcionalidades si fuese necesario.

Interfaz

- Deberá visualizarse de manera correcta en los navegadores Mozilla Firefox, Internet Explorer, Safari, Opera y Google Chrome.

Interfaces de software

- Las búsquedas tendrán una vista amigable para el usuario.

Interfaces de comunicación

- La comunicación entre una PC cliente y el servidor Web será realizada a través del protocolo HTTPS.

Seguridad

- La autenticación de los usuarios y la asignación de permisos basados en roles garantizará que las funcionalidades de la aplicación se muestren de acuerdo al tipo de usuario que esté activo.
- Brindar protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.
- Ofrecer advertencias a los usuarios antes de realizar acciones irreversibles, tal es el caso de las eliminaciones.

2.3.3 Técnicas para la captura de requisitos

La captura de requisitos es la actividad mediante la cual el equipo de desarrollo de un sistema de software extrae de cualquier fuente de información disponible, las necesidades que debe cubrir el mismo. Por la complejidad que esto implica, los especialistas en estos temas han trabajado desde hace años en desarrollar técnicas que permitan hacer este proceso de una forma más eficiente y precisa. La captura de requisitos tiene como objetivo proporcionar el mecanismo adecuado para entender lo que el cliente quiere, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable y especificarla sin ambigüedades. En la identificación y captura de requisitos para el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el Centro de Gobierno Electrónico (CEGEL) se emplearon las siguientes técnicas:

Entrevista

Es una técnica para recopilar información a partir de un intercambio directo entre personas o grupos. Esta fue la primera técnica empleada para la captura de los requisitos con los que debía cumplir el Sistema de Apoyo a la Vigilancia Tecnológica en el Centro de Gobierno Electrónico (CEGEL).

Tormenta de ideas

Es una técnica de grupo para generar ideas. Permitió generar una gran variedad de puntos de vistas del problema y a formularlos de diferentes formas facilitando la concepción del Sistema de Apoyo a la Vigilancia Tecnológica en el Centro de Gobierno Electrónico (CEGEL), sobre todo al comienzo del proceso de captura de los requisitos cuando estos estaban todavía ambiguos.

Sistemas existentes

Esta técnica consiste en analizar distintos sistemas ya desarrollados que estén relacionados con la propuesta de software a implementar, en este caso el Sistema de Apoyo a la Vigilancia Tecnológica en el Centro de Gobierno Electrónico (CEGEL). Permitió analizar módulos, arquitectura y diseño implementados en otras soluciones de Sistemas de Vigilancia Tecnológica.

2.3.4 Técnicas para la validación de los requisitos

Revisiones Técnicas Formales (RTF): Esta es una actividad de garantía de calidad de los sistemas. Son reuniones de un personal (ingenieros del sistema, clientes, usuarios, y otros que examinan la especificación del sistema), con el objetivo de descubrir errores al principio para que no se propaguen al siguiente paso del proceso de desarrollo de software. Su aplicación permite detectar errores en: contenido, interpretación, información incompleta, inconsistencias, requisitos contradictorios, imposibles e inalcanzables, deficiencias y ambigüedades, que luego no se conviertan en defectos después de la entrega del software (Pressman, 2005).

Se realizó una revisión a los requisitos por parte del equipo de desarrollo y por el cliente. En la misma el cliente quedó satisfecho con el trabajo efectuado por el equipo de desarrollo, sin embargo se añadieron 4 nuevos requisitos funcionales, aprobándose finalmente las funcionalidades del sistema.

2.3.5 Métricas para la validación de los requisitos

Estabilidad de los requisitos: El objetivo de esta métrica es medir la estabilidad de los requisitos y asegurar su adecuación antes de pasar al próximo flujo de trabajo. Se considera que los requisitos son estables cuando no existen adiciones o supresiones en ellos que impliquen modificaciones en las funcionalidades principales de la aplicación (Pressman, 2005).

La estabilidad de los requisitos se calcula como: $ETR = [(RT - RM) / RT] * 100$

Dónde:

ETR: valor de la estabilidad de los requisitos.

RT: total de requisitos definidos.

RM: número de requisitos modificados, que se obtienen como la sumatoria de los requisitos insertados, modificados y eliminados.

Esta métrica ofrece valores entre 0 y 100. El mejor valor de **ETR** es el más cercano a 100 porque mostrará que no se están realizando cambios sobre los requisitos, son estables y, por tanto, es confiable trabajar el análisis y diseño sobre ellos (Pressman, 2005).

Se identificaron un total de 25 requisitos funcionales, de los cuales 4 requisitos fueron sometidos a modificaciones, por tanto:

$$ETR = [(25 - 4) / 25] * 100$$

$$ETR = [21/25] * 100$$

$$ETR = [0.84] * 100$$

$$ETR = 84 = 84\%$$

Resultado: El 84% de los requisitos son estables por tanto son confiables por lo que se puede pasar al siguiente paso del proceso del software.

2.4 Fase de Exploración

La metodología de desarrollo de software XP comienza con la fase de exploración en la que se plantean las historias de usuario propuestas por el cliente, siendo el mismo parte del equipo de desarrollo. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología, y concluye cuando el cliente esté convencido de que se poseen las suficientes historias de usuario para realizar una primera entrega con calidad, y los programadores de que han hecho una buena estimación de tiempo y esfuerzo (Beck, y otros, 2000).

2.4.1 Historias de Usuario

Las historias de usuario son la técnica utilizada en la metodología de desarrollo XP para especificar los requisitos del software. Se describen brevemente las características que la aplicación debe tener desde la perspectiva del cliente. El contenido de estas debe ser concreto y sencillo. No es más que un acuerdo que el cliente y los desarrolladores conversarán juntos acerca de una funcionalidad. En la fase de exploración se identificaron 10 historias de usuario, de ellas se describen dos a continuación, para consultar las restantes ver el documento Anexo I - Historias de Usuario.

Tabla 4: Historia de Usuario Nro. 1

Historia de Usuario	
Número: 1	Nombre de la Historia de Usuario: Autenticar usuario
Usuario: Todos los usuarios	Iteración asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 1 semana
Riesgo en desarrollo: Bajo	Puntos reales: 1 semana
Descripción: El usuario introduce sus datos (nombre de usuario y contraseña) para acceder a las distintas opciones del sistema según su rol.	
Observaciones: Un usuario puede ser Administrador, Publicador, Especialista de CEGEL y Usuario Externo.	

Tabla 5: Historia de Usuario Nro. 2

Historia de Usuario	
Número: 2.	Nombre de la Historia de Usuario: Gestionar usuario
Usuario: Administrador	Iteración asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 1 semanas

Riesgo en desarrollo: Alta	Puntos reales: 1 semanas
Descripción: El sistema debe permitir al usuario con rol Administrador realizar todas las operaciones sobre los usuarios. Las operaciones son Adicionar Usuario, Buscar Usuario, Editar Usuario, Eliminar Usuario y Listar Usuario.	
Observaciones:	

2.5 Planificación de la Entrega

En esta fase el cliente establece la prioridad de cada historia de usuario y los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días. Las estimaciones de esfuerzo asociado a la implementación de las historias de usuario la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación.

Por otra parte, el equipo de desarrollo mantiene un registro de la “velocidad” de desarrollo, establecida en puntos por iteración, teniendo en cuenta principalmente la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración. La planificación se puede realizar apoyándose en el tiempo o el alcance del proyecto. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. Al planificar según el alcance del sistema, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación (Beck, y otros, 2000).

2.5.1 Estimación de esfuerzo por Historias de Usuario

Para el desarrollo de la aplicación propuesta en este trabajo se efectuó una estimación del esfuerzo para cada una de las historias de usuario identificadas, obteniéndose los resultados que se exponen en la presente tabla:

Tabla 6: Estimación de esfuerzo por Historias de Usuario

No.	Historias de Usuario.	Puntos Estimados
1	Autenticar Usuario.	1

2	Gestionar Usuario	1
3	Gestionar Roles	1
4	Gestionar Tesis de Grado	2
5	Gestionar Comentarios	1
6	Gestionar Noticias	1
7	Clasificar Tesis de Grado	1
8	Generar Boletín	2
9	Generar Notificaciones	2
10	Interactuar con Buscadores Genéricos	1

2.5.2 Plan de Iteraciones

Luego de identificar cada una de las historias de usuario presentes en la aplicación y estimar el esfuerzo dedicado al desarrollo de estas, se procede a la planificación de la etapa de implementación. Se decidió llevar a cabo la implementación del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL en 3 iteraciones, detalladas a continuación:

Iteración 1

En la iteración número uno se realizarán las historias de usuarios elegidas. Se añadirán las funcionalidades necesarias para gestionar la autenticación y autorización de los usuarios y las funcionalidades relativas a la gestión de roles, las tesis de grado y los comentarios del sitio. La realización de esta iteración dará una primera impresión de cómo se verá el sistema.

Iteración 2

La implementación de esta iteración tendrá como resultado una visión más general del sistema, al desarrollarse las funcionalidades para gestionar las noticias y clasificar las tesis de grado.

Iteración 3

Con esta iteración se finalizará la aplicación, al implementarse las historias de usuario concernientes a generar boletines, generar notificaciones e interactuar con buscadores genéricos como *Google*, *Bing* o *Yahoo Search*.

2.5.3 Plan de duración de las iteraciones

El plan de duración de cada una de las iteraciones se construye como parte del ciclo de vida de un proyecto que utiliza la metodología de desarrollo XP, teniendo presente los equipos de desarrollo con que se cuenta, en este caso se realiza para el único equipo de desarrollo que se posee. Este plan

contiene cada una de las historias de usuario que serán implementadas a lo largo de cada iteración y su duración. En la siguiente tabla se muestra el plan de iteraciones diseñado para el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL.

Tabla 7: Plan de Iteraciones

Iteración	No.	Historias de Usuario a Implementar	Duración Total
1	1	Autenticar Usuario.	3 Semanas.
	2	Gestionar Usuario	
	3	Gestionar Roles	
	4	Gestionar Tesis de Grado	
	5	Gestionar Comentarios	
2	6	Gestionar Noticias	1 Semana.
	7	Clasificar Tesis de Grado	
3	8	Generar Boletín	4 Semanas.
	9	Generar Notificaciones	
	10	Interactuar con buscadores genéricos	

2.5.4 Plan de entregas

El plan de entregas es el compromiso final del equipo de desarrollo con el cliente. En este plan, la estimación es un aspecto importante del desarrollo de un proyecto de software y es por ello que resulta de vital importancia tener claros los requerimientos del cliente, el estilo de trabajo del equipo de desarrollo y el tiempo del que dispone el cliente para tener la solución en sus manos. A continuación se muestra el plan de entregas trazado para el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL.

Tabla 8: Plan de Entregas

Iteración.	Fecha de inicio	Fecha de entregas
------------	-----------------	-------------------

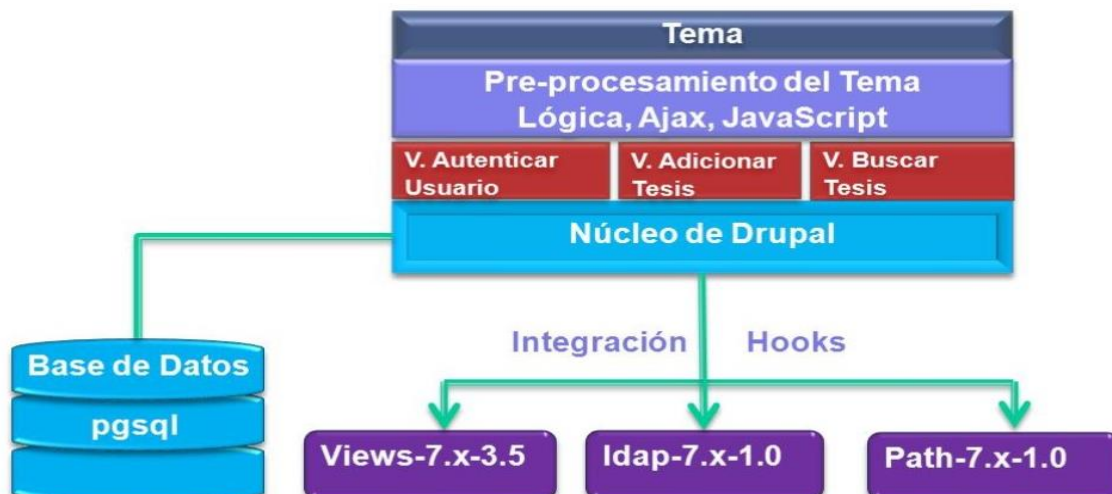
1	30 de marzo	13 de Abril.
2	13 de Abril	20 de Abril.
3	20 de Abril	25 Mayo

2.6 Diseño del Sistema Propuesto

Siguiendo las guías de la metodología XP, debe aparecer un diseño que describa qué clases hay y cómo interactúan. La metodología XP no requiere la representación del sistema mediante diagramas de clases utilizando el Lenguaje Unificado de Modelado (UML); en su lugar usa una técnica para representar clases, esta es la que usa tarjetas CRC. No obstante el uso de diagramas UML puede aplicarse siempre y cuando influyan en el mejoramiento de la comunicación, no sea un peso su mantenimiento, no sean extensos y se enfoquen en la información importante. Con Drupal las tarjetas CRC son un tanto innecesarias, a tal punto que no se aplican, ya que el mismo está compuesto por módulos y no por clases. Estos módulos son implementados de manera independiente para resolver una o varias funcionalidades. Después de implementados son añadidos al sistema y no colaboran con otros módulos debido a la independencia entre ellos y al fin con que se implementan. La programación de estos módulos es una programación estructurada o por funciones.

2.6.1 Arquitectura del sistema

La propuesta de solución va a estar basada en la arquitectura de Drupal, la cual se caracteriza por ser una arquitectura modular que permite ampliar sus funcionalidades a través de unos métodos uniformes de desarrollo e integración de nuevos módulos. En última instancia un módulo consiste en un conjunto de archivos con código PHP, que utiliza la arquitectura y las APIs de Drupal para incorporar nuevas características funcionales al sitio web (Gil, 2012).



Drupal no considera los sitios web como un conjunto de páginas interrelacionadas entre sí, sino que estructura el contenido en una serie de elementos básicos como nodos, módulos, bloques, menús, permisos de usuario y plantillas.

El CMS Drupal se encarga de ejecutar los módulos solo cuando se necesitan, y para ello es necesario que cada módulo tenga un medio de comunicarle qué hace y cuándo debe hacerse. Esto se realiza mediante un conjunto de funciones llamadas *hooks* o ganchos. Los *hooks* van a permitir incorporar simples y complejas funcionalidades al CMS Drupal.

Este gestor de contenidos permite asociarse además con los principales servidores de bases de datos del mundo, en este caso, el sistema estará asociado con PostgreSQL.

Además de los módulos, Drupal está integrado también por temas. Un tema es un conjunto de archivos que permiten cambiar el aspecto gráfico del sitio, estos archivos definen la capa de presentación del sitio. Con los temas, Drupal separa los contenidos de la presentación o diseño gráfico con que se muestran. Cada elemento es tratado de forma independiente, aplicándole su propia plantilla.

2.6.3 Patrones de diseño

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el diseño de software. En otras palabras, brindan una solución ya probada y documentada a problemas de diseño de software que están sujetos a contextos similares. Los patrones de diseño se clasifican en tres tipos (Tedeschi, 2014):

- Patrones Creacionales: Inicialización y configuración de objetos.
- Patrones Estructurales: Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes.
- Patrones de Comportamiento: Más que describir objetos o clases, describen la comunicación entre ellos.

En el CMS Drupal siguiendo una perspectiva orientada a objetos se pueden identificar algunos patrones de diseño, tal es el caso de los patrones GOF.

Patrones GOF (Gang of Four)

Los patrones *GOF* describen soluciones simples a problemas específicos en el diseño de software orientado a objetos (Guerrero, y otros, 2013). En el CMS Drupal se hace uso de varios de estos patrones *GOF*, los cuales se clasifican en:

- Patrones Creacionales

- *Singleton*
- Patrones Estructurales
 - *Bridge*
 - *Decorator*
- Patrones de Comportamiento
 - *Chain of Responsibility*
 - *Observer*
 - *Command*

Singleton (Instancia única)

En la Programación Orientada a Objetos (POO) la esencia del patrón *Singleton* consiste en tener una sola instancia de un objeto disponible para toda la aplicación que la contiene. Provee una instancia global permitiendo que otros objetos accedan a esta única instancia.

En Drupal, si se piensan los módulos como objetos, los cuáles pueden ser pensados como una clase con una única instancia entonces se está en presencia de este patrón. En general lo que diferencia un módulo en Drupal de otro es el conjunto de funciones que este contiene, garantizando así la existencia de una única instancia para un módulo y la creación de un mecanismo de acceso global único a dicho módulo.

Decorator (Decorador)

En la POO el patrón *Decorator* se utiliza para extender la funcionalidad de un objeto de forma dinámica sin tener que cambiar la fuente de la clase original o mediante la herencia. Esto se logra mediante la creación de un contenedor del objeto al que se refiere como decorador alrededor del objeto real.

En Drupal se puede identificar este patrón si se piensa a un nodo como un objeto y haciendo uso de diversos *hooks* y funciones, entre ellos *hook_node_load ()*, *hook_node_view ()* y *node_invoke ()* los cuáles permiten a los módulos acciones arbitrarias para extender su comportamiento sin la necesidad de la creación de subclases o subnodos ampliando la variedad de comportamientos para ser añadidos a los nodos solamente mediante la API de éstos.

Observer (Observador)

El patrón *Observer* define en la POO una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.

En Drupal hay hooks que permiten que los módulos se registren como observadores de los objetos. Un ejemplo se ve cuando se realiza una modificación a un vocabulario en el sistema de taxonomía de Drupal, el *hook hook_taxonomy_vocabulary_update()* se llama en todos los módulos que lo implementan. Al implementar el hook, los módulos se registran como observadores del objeto vocabulario.

Bridge (Puente)

El patrón *Bridge* en la POO promueve la separación de la interfaz de una abstracción de su implementación. En general, el término abstracción se refiere al proceso de identificar el conjunto de atributos y el comportamiento de un objeto que es específico para un uso particular. Este punto de vista específico de un objeto puede ser diseñado como un objeto separado omitiendo atributos y comportamientos irrelevantes. El objeto resultante en sí mismo puede ser referido como una abstracción. Tenga en cuenta que un objeto dado puede tener más de una abstracción asociada, cada uno con un uso distinto.

En Drupal la capa de abstracción de la base de datos es implementada de una manera similar, imitando el patrón *Bridge*, no realizando llamadas directas a la base de datos, sino que se hacen a través de funciones genéricas definidas por la capa de abstracción, funcionando ésta como puente. Estas funciones serán siempre iguales, independientemente del gestor de bases de datos utilizado.

Chain of Responsibility (Cadena de responsabilidades)

Este patrón en la POO permite establecer una cadena de objetos receptores a través de los cuales se pasa una petición formulada por un objeto emisor, proporciona a más de un objeto la capacidad de atender una petición, para así evitar el acoplamiento con el objeto que hace la petición. Se forma entre los objetos una cadena, en la cual cada objeto o satisface la petición o la pasa al siguiente.

En el sistema de menús de Drupal se puede identificar el patrón de cadena de responsabilidades. Cuando el usuario visita una página web, el sistema debe determinar, a partir de la URL, qué módulo es el encargado de su gestión y presentación, siguiendo así un flujo de trabajo en cadena comprobando en todos los módulos activos del sitio a qué módulo pertenece cada solicitud de página, buscando las implementaciones de *hook_menu()*. Una vez encontrado el módulo, este le facilitará al sistema una función de retorno o *callback*, que es la encargada de generar el contenido de la página.

Command (Comando)

Este patrón permite solicitar una operación a un objeto sin conocer realmente el contenido de esta operación ni el receptor real de la misma. Para ello se encapsula la petición como un objeto, con lo que además se facilita la parametrización de los métodos.

Este patrón de diseño es utilizado en Drupal para permitir llevar a cabo la ejecución de ciertas tareas pasando como parámetro el operador, esta es la base fundamental del funcionamiento de los *hooks*.

2.6.3 Seguridad

Actualmente, la mayor parte de los sitios web son multiusuario, por lo que la seguridad y control de los usuarios es un punto clave para garantizar la integridad de la información almacenada. Con esta finalidad Drupal dispone de un registro de usuarios y de roles que permiten especificar qué tareas pueden realizar y a qué contenidos puede acceder cada tipo de usuario. Además establece un sistema de control de acceso basado en roles. La asignación de permisos se realiza a través de los roles del sistema, de esta forma el administrador no tiene que establecer los permisos para cada usuario, sino, se asignan los permisos a un determinado rol y se agrupan los usuarios por roles.

Esta asignación por roles permite controlar a qué funcionalidades el usuario tiene acceso, de manera que la información accedida esté limitada y protegida en dependencia de los distintos niveles de usuarios, obteniendo como resultado un sistema más seguro. Otra forma de brindar seguridad es configurando los permisos de uso de los filtros tales como el “PHP Filter” y el “Full HTML”, de esta manera se logra evitar las inyecciones SQL.

2.7. Conclusiones Parciales

Con la realización de este capítulo quedó definida la propuesta de solución del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL. Se identificaron 10 historias de usuarios y 25 requisitos funcionales los cuales se desglosaron en tareas de ingeniería.

Se identificaron requisitos no funcionales asociados a la Fiabilidad, Eficiencia, Software, Hardware, Soporte, Interfaz y Seguridad; siendo los más importantes para el Sistema de Apoyo a la Vigilancia Tecnológica, la Eficiencia por los tiempos de respuesta esperados, y la Seguridad utilizando las bondades de la plataforma Drupal con las concepciones propias del sistema.

Se utilizaron los patrones GOF: Singleton, Bridge, Decorator, Chain of Responsibility, Observer y Command utilizando una arquitectura modular para el diseño del sistema lo que permitió ampliar las funcionalidades del mismo a través de la integración de nuevos módulos.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA DE APOYO A LA VIGILANCIA TECNOLÓGICA EN EL CENTRO DE GOBIERNO ELECTRÓNICO (CEGEL).

En el presente capítulo se realizará un estudio detallado de la técnica de programación, plataforma y bibliotecas utilizadas en el desarrollo del sistema propuesto. Se observará además cómo es el funcionamiento de un módulo y cómo se procede a la construcción del mismo, se mostrará un ejemplo de algoritmo implementado y se describen también las tres iteraciones desarrolladas durante la etapa de construcción del sistema, presentando cada una de las tareas generadas por cada historia de usuario, así como las pruebas de aceptación confeccionadas.

Al comenzar la implementación de las historias de usuario seleccionadas para cada una de las iteraciones, se lleva a cabo una revisión del plan de iteraciones y se modifica en caso de ser necesario. Teniendo en cuenta la planificación realizada, se llevaron a cabo tres iteraciones de desarrollo sobre el sistema, alcanzándose al finalizar, un producto listo para su puesta en función.

3.1 Estudio Detallado de la Técnica de Programación Utilizada para el Desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL

3.1.1 Estudio de los Módulos de Drupal

Un módulo es la asociación de un conjunto de varias funciones en Drupal que ayudan a ofrecerle mayor funcionalidad a las aplicaciones web. Para crear un módulo en Drupal son necesarios tres tipos de ficheros *PHP*, de los cuales uno es de extensión *.info*, para portar la información pertinente al módulo, como la versión y el autor del módulo; otro fichero es el de extensión *.install*, para crear las tablas necesarias para que el módulo funcione; y por último un fichero de extensión *.module*, que será en donde se escriban funciones que actúen en forma de enganche, al ser llamadas por Drupal a la hora de construir una página web y gestionar el contenido. Ejemplo de estas funciones son: *hook_help*, *hook_perm*, y *hook_menu*. Los módulos se pueden desactivar temporalmente para reducir la carga del servidor. En Drupal existen varios tipos de módulos importantes, como son el módulo de bloque, que da la posibilidad de crear contenido abreviado, que aparecen en la mayoría de los temas en los bloques de la izquierda y la derecha; y están además los módulos de nodo, que generan el contenido de la página del sitio, ejemplo de ello lo constituye la página del foro. A continuación se ofrece una descripción de varios de los módulos del núcleo de Drupal y de los módulos opcionales (Buytaert, 2014).

- **Módulo Comment:** Es el módulo que permite la publicación de comentarios en el sitio, asociados a los contenidos. Provee de las herramientas para poder integrar comentarios en

los tipos de contenido de nuestro sitio y múltiples permisos para poder decidir qué usuarios podrán comentar los contenidos.

- **Módulo File:** Permite que se puedan adjuntar archivos a los contenidos. Define un tipo de campo Archivo que puede ser utilizado en cualquier entidad, por ejemplo: tipos de contenido y usuarios.
- **Módulo Node:** Éste es un módulo imprescindible para el funcionamiento de Drupal y gestiona todas las tareas relacionadas con los nodos. Se encarga de la creación, edición, eliminación, configuración y visualización del contenido del sitio.
- **Módulo Block:** Controla los bloques que se muestran alrededor del contenido principal.
- **Módulo Search (Búsqueda):** Este módulo permite la realización de búsquedas en todo el sitio creado.
- **Módulo User:** Permite que los usuarios se registren, ingresen y terminen la sesión.
- **Módulo Taxonomy:** Activa la categorización del contenido.

Módulos opcionales incluidos en Drupal

- **Módulo Path:** Permite indicar alias a los URLs de Drupal ayudando a mejorar la legibilidad de los mismos para los usuarios. Una página puede tener más de un alias.
- **Módulo Admin Menú (Menú):** Brinda una interfaz para controlar y personalizar el completo sistema de menús que tiene Drupal.
- **Módulo Views:** permite construir páginas dinámicas (llamadas vistas) alimentadas con los contenidos del sitio.
- **Módulo Rules:** permite definir reglas complejas y flexibles para determinar si una acción debe o no ejecutarse al producirse un evento.

3.1.2 Técnicas de programación y creación de módulos con Drupal

Como ya se ha explicado anteriormente para comenzar a crear un módulo es necesario crear un archivo *PHP*, y guardarlo con el patrón *nombre_módulo.module*, otro con *nombre_módulo.install*, este sería para crear las tablas en las bases de datos y un último archivo *nombre_módulo.info* para ofrecer información acerca del módulo. Todas las funciones del módulo que son empleadas por Drupal, siguen la siguiente sintaxis: se nombran como *{hook} _ {nombre_módulo}*, donde "hook" es una función predefinida llamada prefijo. Y Drupal llamará a esas funciones para conseguir datos específicos, teniendo así estos nombres bien definidos. Lo primero que se debe incluir al archivo *.module* debe ser el siguiente fragmento de código (Buytaert, 2014).

```
<?php /* $Id$ */
```

Según el Estándar de Código de Drupal se puede omitir la etiqueta de cierre `?>`, la cadena `Id` permitirá mantener un seguimiento del número y fecha del módulo cuando sea convertido en CVS. Esto significa que cada vez que se modifique este módulo el controlador de versiones de Drupal le asignará un valor distinto al módulo y una fecha de modificación, permitiendo tener un control total sobre los cambios realizados al código y la persona que los realizó. El formato general que debe tener el archivo `.info` es el siguiente:

`; Id`

`name` = Nombre del módulo.

`description` = "Una descripción de lo que hace el módulo."

`dependencies` = "Módulos de los que depende."

A continuación se muestran los ganchos (hooks) más importantes, que se utilizan a la hora de crear un nuevo módulo.

- **Hook_help:** Define temas de ayuda para que sea más fácil el uso del módulo que lo implemente.
- **Hook_access:** Limita restricciones de acceso a los nodos.
- **Hook_perm:** Se especifican quiénes tienen permisos del módulo. Se define quién puede acceder al contenido del sitio o administrar el módulo.
- **Hook_menu:** La función `hook_menu` es llamada siempre por Drupal antes de crear las páginas, antes de mandarlas al navegador, y en ella pueden indicarse los puntos concretos en los que el módulo interactuará con la web.
- **Hook_view:** Es un gancho usado por módulos de nodo. Esto permite a un módulo definir un método de mostrar los nodos.
- **Hook_form_alter:** Permite hacer alteraciones a cualquier formulario antes que sea renderizado.
- **Hook_form:** Es llamado cuando se necesita crear o editar algún artículo y devuelve el título de nodo, el área de texto, y otro campo específico del tipo de nodo.
- **Hook_submit:** Se usa para las acciones que deben ocurrir cuando el módulo es guardado. Y también es utilizado después de que la validación ha tenido éxito y antes de insertar o actualizar.
- **Hook_validate:** Esto es un gancho utilizado por módulos de nodo. Es utilizado para permitir al módulo verificar que el nodo se encuentra en un formato válido para añadirse al sitio.

- **Hook_delete:** Es un gancho empleado por módulos de nodo. Permite al módulo tomar medidas cuando un nodo es eliminado de la base de datos, por ejemplo, borrando la información de tablas relacionadas.
- **Hook_update:** Indica cuando un nodo se actualiza. Es utilizado para permitir al módulo tomar acciones cuando un nodo es actualizado en la base de datos.
- **Hook_settings:** Para que el administrador pueda configurar lo que se va a mostrar, se crea la página de configuración. Casi siempre el administrador es quien tiene acceso a esta página.
- **Hook_search_item:** Este módulo permite modificar los resultados de una búsqueda.
- **Hook_search:** Permite a un módulo realizar búsquedas sobre el contenido que este define cuando una búsqueda de sitio es realizada.

3.2 Estándares de Codificación

El CMS Drupal cuenta con varios estándares de codificación que se exponen a continuación (Gil, 2012).

Indentación

La indentación consiste en insertar espacios en blanco o tabuladores en determinadas líneas de código para facilitar su comprensión. En programación se utiliza la indentación para anidar elementos. En Drupal se debe indentar con 2 espacios, nunca con tabuladores. Además, no se debe dejar espacios en blanco al final de cada línea.

Etiquetas de apertura y cierre de PHP

Cuando se esté escribiendo en *PHP*, siempre se deben utilizar las etiquetas `<?php` y `?>`, y en ningún caso la versión corta `<? y ?>`. En general se omite la etiqueta de cierre de *PHP* (`?>`) al final de los archivos `.module` y `.inc`. Esta convención evita que se puedan quedar olvidados espacios no deseados al final del archivo (después de la etiqueta de cierre `?>`), que serían identificados como salida *HTML* y podrían provocar un error muy típico, *"Cannot modify header information - headers already sent by..."*. Por tanto, la etiqueta de cierre final del archivo (`?>`) es opcional en Drupal. No hay que confundir esto con el uso normal del lenguaje *PHP* en archivos que también contienen *HTML* (por ejemplo los archivos de plantilla `.tpl.php`), donde cada fragmento de *PHP* debe llevar sus correspondientes etiquetas de apertura y cierre, para diferenciarlo del código *HTML*.

Operadores

Los operadores binarios, que se utilizan entre dos valores, deben separarse de estos valores, a ambos lados del operador, por un espacio. Por ejemplo, `$cantidad = 3`, en lugar de `$cantidad=3`. Esto se aplica a operadores como `+`, `-`, `*`, `/`, `=`, `==`, `!=`, `>`, `<`, `.` (Concatenación de cadenas), `=+`, `=-`. Los

operadores unarios como ++, -- no deben tener separación, un ejemplo de operador unario es: `$numero++`.

Uso de comillas

Se pueden usar tanto las comillas simples ('cadena') como las comillas dobles ("cadena") para delimitar las cadenas de caracteres. Las comillas dobles son necesarias si se desean incluir variables dentro de las cadenas de texto. Por ejemplo, "`<h1> $title </h1>`". También se recomienda el uso de comillas dobles cuando el texto puede incluir alguna comilla simple.

Uso de punto y coma (;) en código PHP

Aunque *PHP* permite escribir líneas de código individuales sin el terminador de línea (;), por ejemplo `<?php print $title ?>`. En Drupal es siempre obligatorio: `<?php print $title; ?>`.

Estructuras de control

Con respecto a las estructuras de control, para utilizarlas en la implementación de módulos en Drupal hay que tener en cuenta las siguientes normas:

- Debe haber un espacio entre el comando que define la estructura (*if*, *while*, *for*, *switch*.) y el paréntesis de apertura. Esto es así para no confundir las estructuras de control con la nomenclatura de las funciones.
- La llave de apertura { se situará en la misma línea que la definición de la estructura, separada por un espacio.
- Se recomienda usar siempre las llaves {} aún en los casos en que no sea obligatorio su uso (una sola "línea" de código dentro de la estructura de control).
- Las estructuras *else* y *elseif* se escribirán en la línea siguiente al cierre de la sentencia anterior.

Funciones

Los nombres de las funciones deben estar escritos en minúsculas y las palabras separadas por guion bajo. Además, se debe incluir siempre como prefijo el nombre del módulo, tema, para evitar así duplicidad de funciones.

En su declaración, después del nombre de la función, el paréntesis de inicio de los argumentos debe ir sin espacio. Cada argumento debe ir separado por un espacio, después de la coma del argumento anterior. Por ejemplo: `function forum_help($path, $arg) {}`.

En la llamada a la función se aplican las mismas reglas anteriores con respecto a los parámetros, como se muestra en el siguiente ejemplo: `$field = field_info_instance('node', 'taxonomy_forums', $node->type);`

Arrays

Los valores dentro de un *array* (o matriz) se deben separar por un espacio (después de la coma que los separa). El operador => debe separarse por un espacio a ambos lados.

Cuando la línea de declaración del *array* supera los 80 caracteres, cada elemento se debe escribir en una única línea, indentándolo una vez (2 espacios). En este último caso, la coma de separación del último elemento también se escribirá, aunque no existan más elementos. De esta forma se evitan errores al añadir nuevos elementos al vector.

```
$vector1 = array(1, 2, 'clave' => 'valor');

$vector2 = array(
  'forum' => 'forol',
  'template' => 'forums',
  'arguments' => array('tid' => NULL, 'topics' => NULL),
  'size' => 128,
);
```

Figura 3: Ejemplo de Indentación en Arrays

Constantes

Los nombres de las constantes deben escribirse en mayúsculas, con guiones bajos para separar palabras. Al igual que ocurre con las funciones, los nombres de las constantes deben tener como prefijo el nombre del módulo (o tema) en el que se utilizan, para evitar errores de duplicidad de constantes. Este prefijo también se escribirá en mayúsculas. Este es un ejemplo:

```
define('DRUPAL_MINIMUM_PHP', '5.2.4');
```

Variables globales

Aunque el uso de variables globales está desaconsejado, en caso de necesitarse, éstas se declararán utilizando un guión bajo inicial, seguido del nombre del módulo o tema y otro guión bajo antes del nombre de la variable. Ejemplo de ello: *global \$_forum_numero_foros;*

Nombres de módulos

Como norma general, el nombre de un módulo nunca debería incluir guiones bajos, aunque se componga de varias palabras. De esta forma será más fácil identificar el módulo al que pertenece una función, ya que el prefijo o nombre del módulo será todo aquello que esté antes del primer guión bajo. Por ejemplo, es aconsejable utilizar *mimodulo* en lugar de *mi_modulo*. Esta regla no es obligatoria, y es muy común encontrar, entre los módulos contribuidos, nombres conteniendo guiones bajos.

Nombres de archivos

Los nombres de archivos deben escribirse siempre en minúsculas. La única excepción son los archivos de documentación, que tendrán extensión *.txt* y el nombre en mayúsculas. Por ejemplo, *README.txt*, *INSTALL.txt*.

URLs de ejemplo

Por convención, siempre que haya que indicar una URL de ejemplo, se debe utilizar *"example.com"*. Por ejemplo: *http://example.com/node/add/tesis*.

Idioma

Drupal utiliza el idioma inglés como base para todo el sistema. Es por ello que todos los módulos deberían escribirse en inglés, tanto su código (nombres de funciones, variables) como las cadenas de texto que se mostrarán al usuario. Para que estas cadenas sean traducibles se utilizó la función de traducción `t()`.

3.1.4 Análisis de posibles componentes, módulos e implementaciones ya existentes que pueden ser reutilizados por los autores de la presente investigación

A continuación se relaciona un conjunto de componentes, módulos e implementaciones ya existentes que pueden ser reutilizados así como sus características fundamentales (Buytaert, 2014).

- **Administration Menu:** El módulo de Administración de menú proporciona una interfaz de administración independiente para Drupal. Es una ayuda para los usuarios principiantes de Drupal procedentes de otros CMS, un ahorro de tiempo para los administradores del sitio de Drupal, y definitivamente una necesidad para los desarrolladores de Drupal. El módulo está basado en CSS atractivo y exclusivamente en la parte superior del sitio web.
- **Admin Role:** Este módulo sirve de ayuda para mantener una función de administrador que tiene todos los permisos disponibles. Por defecto, Drupal sólo tiene un súper usuario y este módulo ayuda a mejorar esto de forma radical.
- **Automated Logout:** Este módulo proporciona un administrador de sitio que desconecta a un usuario tras pasar un periodo de inactividad especificado. Es altamente personalizable e incluye "las políticas del sitio" por el papel en la aplicación de cierre de sesión.
- **Counter:** El módulo contador cuenta el número de usuarios en el sitio.

Características:

- Cuenta en la web la entrada de usuarios por día, minuto, incluso segundos.
- Cuenta los usuarios registrados y no registrados.
- Emite estadísticas de visitantes por días, semanas, meses y años.

- **LDAP Integration:** Módulo que permite la autenticación de un sitio contra un dominio. Este es válido para cualquier dominio, además tiene acceso pleno o restringido a toda la información de la base de datos del dominio.
- **Filter:** La lista de módulos puede llegar a ser bastante grande cuando se trata de un sitio grande. Este proporciona la capacidad de encontrar rápidamente el módulo que se está buscando sin tener que depender de la función de búsqueda de los navegadores. Cuando se habilita mediante la configuración del filtro del módulo de página, se aplica un nuevo tema de diseño del módulo. Este diseño proporciona una ficha para cada paquete, así como una ficha donde se muestran todos los módulos en orden alfabético.
- **Pathauto:** El módulo genera automáticamente alias de ruta para los diversos tipos de contenido (nodos, categorías y usuarios), sin necesidad de que el usuario especifique manualmente el alias de ruta. Esto le permite obtener los alias como / categoría / mi-nodo-title.html en lugar de node/123. Los alias se basan en un "patrón" que el administrador del sistema puede controlar.

3.1.5 Descripción del módulo `gestionar_usuarios` y del algoritmo más complejo en su desarrollo

En la elaboración del sistema fue necesario implementar un módulo llamado *gestionar_usuarios*. Este módulo permite la autenticación de los usuarios y una vez autenticado permite ir a su perfil de usuario. El módulo fue implementado con lenguaje *PHP* y siguiendo el paradigma de una programación estructurada y no orientada a objetos. Las funcionalidades implementadas tienen como objetivo permitir a los usuarios una autenticación en el sistema de forma fácil.

El módulo está compuesto por tres tipos de archivos: *gestionar_usuarios.info*, que muestra la información general del módulo; *gestionar_usuarios.install*, donde se crean las tablas necesarias y por último está *gestionar_usuarios.module*, que sería el archivo principal donde se procede a implementar todas las funciones.

En el capítulo anterior se explica por qué cuando se utiliza la metodología XP es innecesario usar tarjetas CRC si se desarrolla con Drupal. La no necesidad de estas está dada en que la implementación en Drupal es mediante módulos y no por clases. Estos módulos se implementan mediante funciones siendo así una programación estructurada. Teniendo en cuenta esto y siguiendo la estrategia utilizada por XP en sus tarjetas CRC, se conforma una tabla Módulo-Responsabilidad-Colaboración MRC que nos muestra los datos principales del módulo y la colaboración de este con los demás en del sistema.

Tabla 9: Módulo-Responsabilidad-Colaboración

Módulo: gestionar_usuarios.	
Responsabilidad	Colaboración
Permitir la autenticación a los usuarios en el sistema.	Módulo: <i>Block</i> .
Permitir revisar el perfil de usuario propio a cada usuario autenticado.	Módulo: <i>User</i>
Permitir cerrar la sesión de los usuarios autenticados.	

Ejemplo de Algoritmo implementado.

```

/**
 * Implements hook_block_view().
 */
function gestionar_usuarios_block_view($delta = '') {
  $block = array();
  switch ($delta) {
    case 'user_data':
//necesitamos acceder al objeto $user del usuario actual
    global $user;
    $block['subject'] = t('Datos de Usuario');
    if (user_is_logged_in()){
      if ($user->picture){
//carga el objeto archivo de la imagen de usuario
        $picture_file = file_load($user->picture);
//obtiene la URL final a partir de la URL en formato URI
        $picture_url = file_create_url($picture_file->uri);}
        $output = '<div class="user_block_registered">';
        if ($user->picture) {
          $output.= '<div id="image">'. l('', 'user/'.
          $user->uid, array('html'=>TRUE)). '</div>';
        }
        $output.= l($user->name, 'user/'. $user->uid);
        $output.= '</div>';
        $output.= '<div id="logout">'.l(t('Cerrar sesión'), 'user/logout').
        '</div>';
      }else{
        $output = '<div class="user_block_anonymous">';
        $output = '<p>Usted no ha iniciado <a href="/user/login"
        title="Iniciar Sesión">sesión</a> o no está; <a href="/user/register"
        title="Regístrame">registrado</a><p>';
        $output.= '</div>';
      }
    }
  }
}

```

```

$block['content'] = $output;
break;
}
return $block;
}

```

3.3 Tareas de Programación

Las tareas de programación se definen con el objetivo de desglosar cada historia de usuario en tareas que serán realizadas por los programadores, proporcionándoles una guía para su desarrollo y cumplimiento. Las tareas de programación son actividades que los programadores conocen que el sistema debe hacer. Deben ser estimables e implementadas entre uno y tres días ideales. La mayoría de las tareas de programación se derivan directamente de las historias de usuario. A continuación se muestran las tareas de programación de la historia de usuario Gestionar Roles. Las restantes tareas de programación pueden ser consultadas en el Anexo II-Tareas de Programación.

Tabla 10: Tarea de Programación #1

Tarea de Programación	
Numero de tarea: 1	Número de historia: 3
Nombre de tarea: Configurar la interfaz para la gestión de roles.	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio: 4/4/2014	Fecha fin: 4/4/2014
Descripción: Se configura la interfaz para gestionar los roles.	

Tabla 11: Tarea de Programación #2

Tarea de Programación	
Numero de tarea: 2	Número de historia: 3
Nombre de tarea: Inserción de los datos referentes a los distintos roles.	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio: 5/4/2014	Fecha fin: 5/4/2014

Descripción: Se insertarán los datos referentes a los distintos roles.

3.4 Diagrama de Despliegue

Un diagrama de despliegue muestra cómo y dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos, y la construcción interna puede ser representada por nodos o artefactos embebidos. Como los artefactos se ubican en los nodos para modelar el despliegue del sistema, la ubicación es guiada por el uso de las especificaciones de despliegue (Sparx Systems Pty. Ltd, 2007)

Teniendo en cuenta lo citado anteriormente, se confeccionó el diagrama de despliegue del Sistema de Apoyo a la Vigilancia Tecnológica el cual se muestra a continuación, en el mismo se expone que el servidor web y el servidor de base de datos se instalarán en la misma PC física, las PCs clientes se comunicarán a través del protocolo *HTTPS* con el servidor web y dicho servidor se comunicará con el servidor de directorios de la Universidad de las Ciencias Informáticas a través del protocolo *LDAP*.

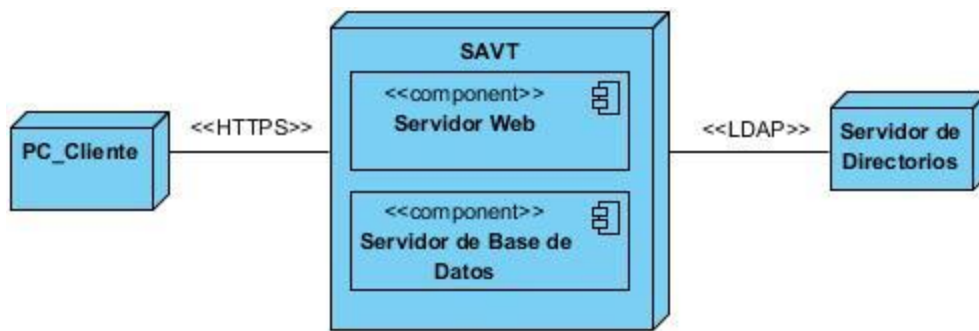


Figura 4: Diagrama de Despliegue

3.5 Estrategia de Prueba

En el proceso de desarrollo de un software, la etapa de prueba es clave a la hora de detectar errores o fallas. Conceptos como estabilidad, escalabilidad, eficiencia y seguridad se relacionan a la calidad de un producto bien desarrollado. Las pruebas muestran que nuestro producto final funciona, cuando se está totalmente convencido de que se han realizado todas las pruebas que pudiesen producir un fallo en el sistema.

Uno de los pilares de la metodología XP es el proceso de pruebas. Esta metodología anima a probar constantemente el software, esto permite aumentar la calidad de los sistemas, reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su solución. También permite aumentar la seguridad y evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones. “Las pruebas unitarias y funcionales son el corazón de XP” (Beck, y otros, 2000).

La metodología XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final (Cohn, y otros, 2004).

3.5.1 Tipo de prueba

Para la validación de los requisitos funcionales se define como tipo de prueba a realizarse: las pruebas funcionales o de aceptación que estarán dentro de la parte de validación. Este tipo de pruebas funcionales están centradas en comprobar que las funcionalidades descritas en el documento de requisitos del sistema se cumplen con la implementación realizada (Oré B., 2009).

3.5.2 Método de prueba

Para la realización de las pruebas funcionales se tiene el método de prueba de caja negra. Este método se refiere a las pruebas que se llevan a cabo sobre la interfaz del software, por lo que los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene. Esta prueba examina algunos aspectos del modelo fundamentalmente del sistema sin tener mucho en cuenta la estructura interna del software.

3.5.3 Técnica de prueba: Partición de Equivalencia o Casos de prueba

Dentro del método de caja negra, la técnica de la Partición de Equivalencia es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el software. Esta divide el dominio de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. El diseño de estos casos de prueba para la partición equivalente se basa en la evaluación de un conjunto de estados válidos o inválidos (clases de equivalencia) para una condición de entrada.

Para garantizar la calidad del sistema se diseñaron pruebas funcionales que son aplicadas a través de pruebas de aceptación. Las pruebas de aceptación son pruebas de caja negra que se crean a partir de las historias de usuario. Durante las iteraciones las historias de usuarios seleccionadas serán traducidas a pruebas de aceptación. En ellas se especifican, desde la perspectiva del cliente, los escenarios para probar que una historia de usuario ha sido implementada correctamente. Una historia de usuario puede tener todas las pruebas de aceptación que necesite para asegurar su correcto funcionamiento. El objetivo final de éstas es garantizar que los requerimientos han sido cumplidos y que el sistema es aceptable. Una historia de usuario no se considera completa hasta que no ha pasado por sus pruebas de aceptación.

A continuación se muestra un ejemplo de caso de prueba de aceptación para la historia de usuario Gestionar Tesis de Grado, para consultar las restantes (Ver: Anexo III – Pruebas de Aceptación).

Tabla 12: Caso de Prueba de Aceptación HU4_P1

Caso de Prueba de Aceptación.	
Código: HU4_P1	Historia de Usuario: Gestionar Tesis de Grado
Nombre: Gestionar Tesis de Grado	
Descripción: Probar que se adicione, se modifique y se elimine correctamente una Tesis de Grado en el sistema.	
Condiciones de Ejecución: La aplicación debe ser ejecutada con privilegios de administración. Debe permitir a un usuario con rol Administrador o Publicador adicionar, modificar y eliminar Tesis de Grado en la base de datos.	
Entrada/Pasos de Ejecución: Un usuario se autentica como Administrador o Publicador y luego intenta adicionar, modificar y eliminar una Tesis de Grado.	
Resultado Esperado: La Tesis de Grado es insertada correctamente en el sistema y gestionada por un usuario autenticado como Administrador o Publicador.	
Evaluación de la Prueba: Satisfactoria	

Tabla 13: Caso de Prueba de Aceptación HU4_P2

Caso de Prueba de Aceptación.	
Código: HU4_P2	Historia de Usuario: Gestionar Tesis de Grado
Nombre: Gestionar Tesis de Grado	
Descripción: Probar que un usuario externo o especialista de CEGEL no pueda gestionar una Tesis de Grado.	
Condiciones de Ejecución: La aplicación debe ser ejecutada en condiciones normales. El sistema no debe permitir que una Tesis de Grado pueda ser adicionada, modificada y eliminada en la base de datos por un usuario externo o especialista de CEGEL.	
Entrada/Pasos de Ejecución: Un usuario sin autenticarse o autenticado como Especialista CEGEL intenta adicionar, modificar y eliminar una Tesis de Grado.	

Resultado Esperado: El sistema muestra un error indicándole al usuario que no tiene los permisos suficientes para gestionar las Tesis de Grado.

Evaluación de la Prueba: Satisfactoria

3.5.4 Resultados de las pruebas de aceptación

Se diseñaron 18 casos de prueba que responden a las funcionalidades del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL. En total fueron detectadas 43 no conformidades en tres iteraciones de prueba.

En la primera iteración se obtuvieron 28 no conformidades. Entre las que se encuentran errores de validación, errores ortográficos, errores de traducción, la ausencia de campos en tipos de contenidos creados y algunos formatos de presentación de contenidos erróneos. Estas no conformidades fueron corregidas.

En la segunda iteración se detectaron 15 no conformidades. Dentro de las que se encuentran: errores ortográficos y errores de traducción. Estas no conformidades también fueron rectificadas.

En la tercera iteración el sistema quedó libre de no conformidades, cumpliendo con los requisitos funcionales especificados. La gráfica que a continuación se presenta, muestra los resultados anteriormente mencionados:

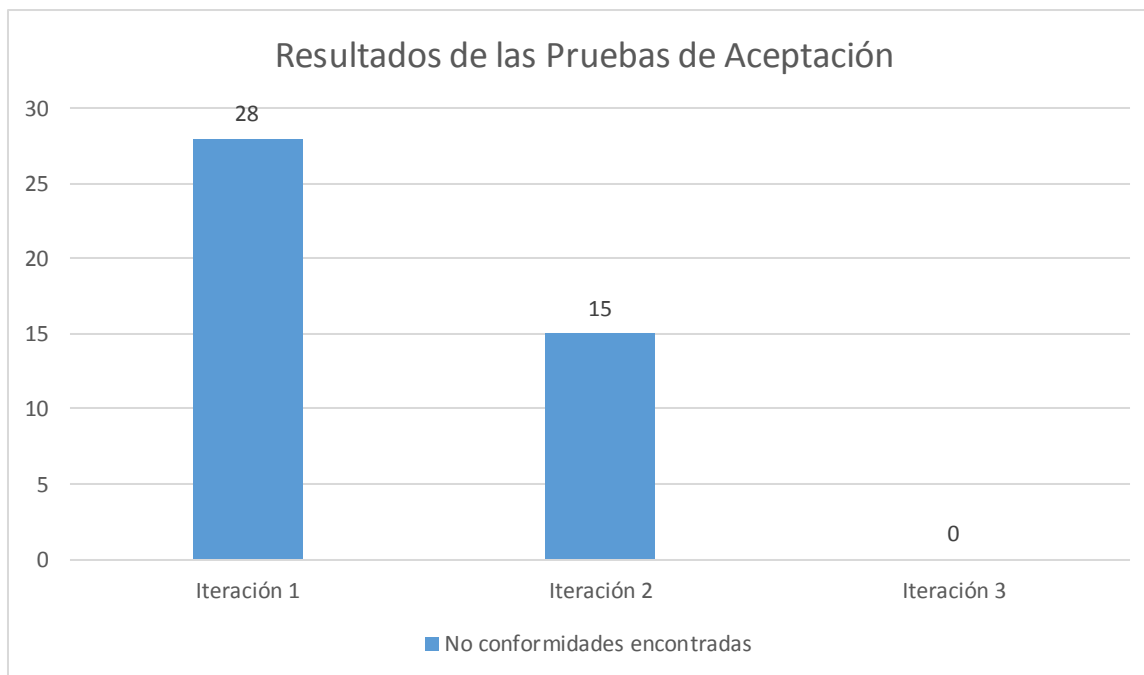


Figura 5: Resultados de las Pruebas de Aceptación

3.5.5 Pruebas de Usabilidad

Las pruebas más relacionadas con mejorar la satisfacción del cliente son las de usabilidad, las cuales son técnicas formales que tienen como objetivo estudiar la usabilidad de una aplicación en un entorno real con usuarios reales. Usabilidad se define como el grado en el que un producto puede ser utilizado por usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un determinado contexto de uso (Vega Prieto, y otros, 2010).

A través de la lista de chequeo diseñada por el Laboratorio Industrial de Pruebas de Software perteneciente a la empresa CALISOFT se realizaron las pruebas de usabilidad al Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL. Esta lista de chequeo cuenta con una serie de preguntas o indicadores que permiten evaluar varios puntos relacionados con la usabilidad de sitios web. La forma de evaluar cada indicador es la siguiente: el mismo se evalúa con “1” en caso de mal (cuando la respuesta al indicador sea “No”) y se le otorga el valor “0” en caso de que el elemento revisado no presente errores (cuando la respuesta al indicador sea “Sí”). Luego se calcula el porcentaje de preguntas evaluadas de bien en función del total y si el resultado es superior al 70 % se considera que la prueba arrojó resultados positivos. A continuación se muestra en una gráfica el resultado de estas pruebas, en las que el cumplimiento de los indicadores de cada área de usabilidad fue mayor al 70%, demostrándose que el Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL es usable.

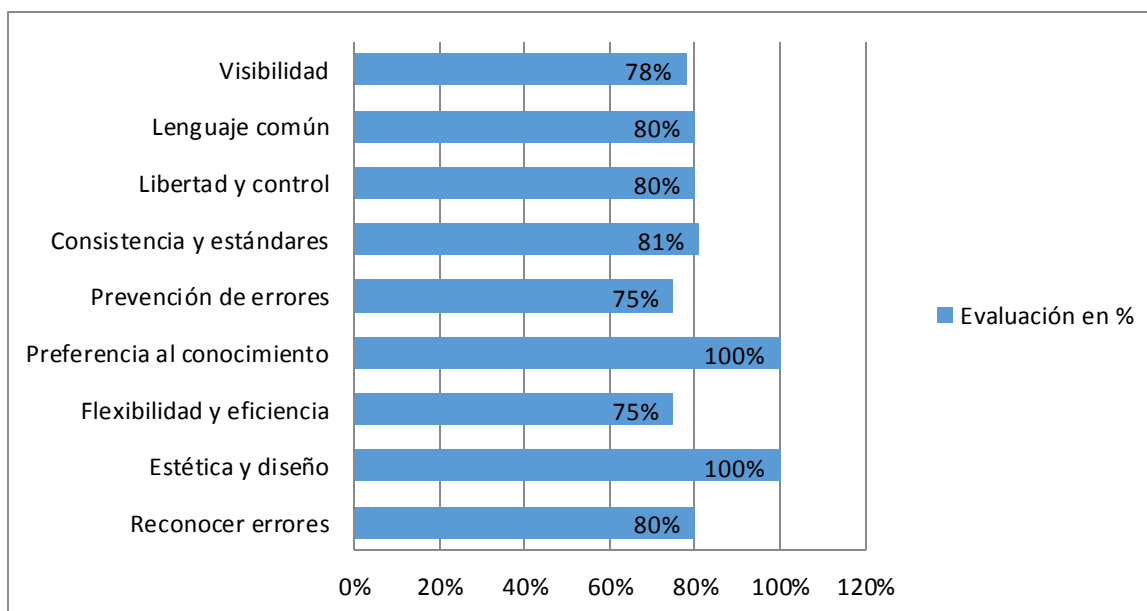


Figura 6: Resultados de las Pruebas de Usabilidad

3.6 Conclusiones Parciales

Con el desarrollo de este capítulo fue posible, a partir de la filosofía de trabajo de Drupal, la reutilización de los módulos *Administration Menu*, *Automated Logout*, *Counter*, *LDAP Integration*, *Filter* y *Pathauto* por su correspondencia con las características funcionales que el mismo presenta y la función objeto de implementación.

La personalización de los módulos *User* y *Views* permitió agregarle características a la gestión de usuarios y la gestión de las Tesis de Grado, utilizando los estándares de codificación definidos para el CMS Drupal y expuestos por los autores.

Los resultados de las pruebas de caja negra arrojaron 43 no conformidades, 28 de ellas en una primera iteración y las 15 restantes en una segunda iteración, en una 3ra iteración el sistema quedó libre de no conformidades. Además el análisis de usabilidad superó el 70 por ciento el cumplimiento de los indicadores, por cada área de usabilidad; por tanto el sistema se considera usable y lo más libre posible de errores desde la vista de los autores.

CONCLUSIONES GENERALES

Con el desarrollo de la presente investigación se arribaron a las siguientes conclusiones:

- El estudio de los procesos de la Vigilancia Tecnológica permitió identificar los servicios más adecuados para la propuesta de solución. La metodología de desarrollo de software y las herramientas seleccionadas para el desarrollo del sistema permitieron una correcta organización e implementación del mismo.
- La planificación y el diseño del sistema permitieron disminuir el tiempo y recursos utilizados en el desarrollo de la solución, se logró un mayor entendimiento de las necesidades del cliente.
- Las pruebas de usabilidad y los resultados obtenidos a través de las pruebas de aceptación con el cliente demostraron que el sistema es usable y está libre de errores.
- Con el desarrollo del Sistema de Apoyo a la Vigilancia Tecnológica en el CEGEL se eliminaron carencias en el proceso de Vigilancia Tecnológica que dificultaban la disponibilidad de información actualizada relacionada con los productos que se desarrollan, de interés para los gestores de proyectos y su equipo de trabajo.

RECOMENDACIONES

- Profundizar en el estudio de otros tipos de Vigilancia Tecnológica acorde a sus objetos de vigilancia para su uso en versiones posteriores del sistema.
- Mantener una cercana información acerca de las actualizaciones del CMS Drupal y todos los módulos instalados en el sitio.
- Adicionar nuevas funcionalidades que permitan brindar otros servicios de información (catálogo de Tesis de Maestría y catálogo de Tesis de Doctorado).

BIBLIOGRAFÍA

- AENOR. 2011.** Norma UNE 166000:2011. [En línea] 2011. <http://www.aenor.es/aenor/normas/normas/fichanorma.asp?tipo=N&codigo=N0046930>.
- Agut, Raul Monferrer. 2001.** Especificación de Requisitos de Software según el estándar IEEE 830. [En línea] 2001. [Citado el: 20 de enero de 2014.] <http://siml.googlecode.com/files/ERS.pdf>.
- Angelozzi, Silvina Marcela y Martín, Sandra Gisela. 2011.** Vigilancia Tecnológica e Inteligencia Competitiva: aportes desde las. [En línea] 2011. [Citado el: 10 de diciembre de 2013.] <http://eprints.rclis.org/16752/1/2011%20Vigilancia%20Tecnol%C3%B3gica%20e%20Inteligencia%20Competitiva%20aportes%20desde%20las%20bibliotecas%20y%20centros%20de%20documentaci%C3%B3n.pdf>.
- Beck, K y Wesley, Addison. 2000.** *Extreme Programming Explained*. 2000.
- Blog, PostgreSQL. 2012.** [En línea] 2012. <http://postgresql-dbms.blogspot.com/p/limitaciones-puntos-de-recuperacion.html>.
- Buytaert, Dries. 2014.** Drupal. [En línea] 2014. <https://drupal.org/documentation/develop>.
- Cohn, M. y Wesley, Addison. 2004.** *User Stories Applied: For Agile Software Development*. 2004.
- Colectivo de Autores. 2012.** Estudio de los sistemas de gestión de contenidos web. [En línea] 2012. http://sunshine.prod.uci.cu/gridfs/sunshine/books/Estudio_de_los_sistemas_de_gestion_de_contenidos_web.pdf.
- Díaz Flores, Mirian Milagros. 2013.** RUP vs XP. [En línea] 2013. <http://www.usmp.edu.pe/publicaciones/boletin/fia/info49/articulos/RUP%20vs.%20XP.pdf>.
- Dlógica. 2013.** Comparativa CMS: Drupal, Joomla y WordPress. *Dlógica*. [En línea] 1 de agosto de 2013. [Citado el: 26 de octubre de 2013.] <http://www.dlogica.com/sites/default/files/resources/blog/DLO-COMPARATIVA-CMS-001.pdf>.
- Dominguez, Abraham. 2008.** Manual Básico de Wordpress. [En línea] 2008. comunidad.cav.uci.cu/index.php?option=com_phocadownload&view=category&download=58:manual-wordpress&id=28:manuales-y-tutoriales&Itemid=56.
- Eguiluz, Javier. 2014.** Librosweb. [En línea] 2014. [Citado el: 3 de abril de 2014.] <http://librosweb.es/xhtml/descargar/pdf/>.
- Fernandez Fuentes, Belén, Pérez Álvarez, Sara y del Valle Gastaminza, Félix. 2009.** Metodología para la implantación de sistemas de vigilancia tecnológica y documental: El caso del proyecto INREDIS. [En línea] 2009. http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S0187-358X2009000300006.

- Font Rodriguez, Marisela y Hernandez Leyva, María Victoria. 2008.** Sistema de Vigilancia Tecnológica del Ministerio de la Informática y las Comunicaciones y D´TIC, el Centro Virtual de Recursos de Información. [En línea] 2008. <http://www.redciencia.info.ve/memorias/ProyProsp/trabajos/e4.doc>.
- García Pérez, Lourdes y Santana Monzon, Dubiel. 2010.** Repositorio Institucional. *Herramienta de software para automatizar la medición de productos de software en los proyectos productivos de la facultad* 9. [En línea] 2010. [Citado el: 12 de 12 de 2013.] http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_03600_10/1/TD_03600_10.pdf.
- Gestiopolis. 2014.** Comunidad virtual de conocimiento en gestión de negocios. [En línea] 2014. <http://www.gestiopolis.com/recursos/documentos/fulldocs/ger/GestionelInnovacion.pdf>.
- Gil, Fran. 2012.** *Experto en Drupal. Nivel Avanzado*. s.l.: Forcontu S.L, 2012. 978-84-939410-2-4.
- Graf, Hagen, Kramer, Jen y Mitova, Milena. 2012.** *Joomla 2.5 Guía para principiantes*. 2012.
- Guerrero, Carlos A, Suárez, Johanna M. y Gutierrez, Luz E. 2013.** Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. *Scielo. Información tecnológica Vol.24*. [En línea] 2013. [Citado el: 28 de febrero de 2014.] http://www.scielo.cl/scielo.php?pid=S0718-07642013000300012&script=sci_arttext.
- Guervos Merelo, Juan Julian. 2005.** Introducción a los sistemas de gestión de contenidos. [En línea] 2005. [Citado el: 25 de noviembre de 2013.] <http://geneura.ugr.es/~jmerelo/tutoriales/cms/>.
- Henst, Christian Van Der. 2001.** Qué es PHP? *Maestros del web*. [En línea] 23 de mayo de 2001. [Citado el: 29 de octubre de 2013.] <http://www.maestrosdelweb.com/editorial/phpintro/>.
- Hispano, Drupal. 2013.** Comunidad de usuarios de Drupal. [En línea] 2013. [Citado el: 28 de octubre de 2013.] <http://drupal.org.es/caracteristicas>.
- Jeffries, Ron. 2010.** [En línea] 5 de Febrero de 2010. <http://www.dosideas.com/noticias/metodologias/822-una-introduccion-a-extreme-programming.html>.
- Joomla. 2014.** Centro de Ayuda de Joomla. *¿Qué es Joomla?* [En línea] 2014. <http://ayuda.joomlaspanish.org/que-es-joomla>.
- Kaisler, Stephen. 2005.** *Software Paradigms*. New Jersey: John Wiley & Sons, 2005. ISBN 0-471-48347-8.
- Kniberg, Henrik. 2007.** Scrum and XP from the Trenches. [En línea] 2007.
- Lara Pérez, Marlon y González Rodriguez, Leover Armando. 2009.** Sistema de Gestión de la Vigilancia Tecnológica en el MIC. [En línea] 2009. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_2375_09/1/TD_2375_09.pdf.

- Letelier, Patricio y Penadés, Maria del Carmen. 2012.** Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). [En línea] 2012.
- Martínez Negrín, Laritza Magdalena. 2012.** Propuesta de un Sistema de Vigilancia Tecnológica para la UCI. [En línea] 2012. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/7934/1/Tdig_0051_12.pdf.
- Mi+d. 2014.** Mi+d.Un lugar para la ciencia y la tecnología. [En línea] 2014. <http://www.madrimasd.org/vigtecnologica/vt-madrimasd/default.asp>.
- MySQL. 2011.** MySQL 5.0 Reference Manual. *Las principales características de MySQL*. [En línea] 2011. [Citado el: 30 de octubre de 2013.] <http://dev.mysql.com/doc/refman/5.0/es/features.html>.
- Naramore, Elizabeth. 2005.** *Beginning PHP5, Apache, and MySQL Web Development*. Canada: Wiley Publishing, 2005. ISBN: 0-7645-7966-5.
- Negrín, Lic. Laritza Magdalena Martínez. 2012.** Propuesta de un Sistema de Vigilancia Tecnológica para la UCI. *Biblioteca UCI*. [En línea] 2012. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/7934/1/Tdig_0051_12.pdf.
- Oliva, Luis. 2008.** Metodologías de desarrollo SAP. [En línea] 20 de septiembre de 2008. [Citado el: 30 de octubre de 2013.] <http://www.luisoliva.blogspot.com/2008/09/metodologas-de-desarrollo-sap.html>.
- Oré B., Ing. Alexander. 2009.** Pruebas funcionales. [En línea] 2009. http://www.calidadyssoftware.com/testing/pruebas_funcionales.php.
- Palacios, Juan. 2011.** Entorno Virtual de Aprendizaje. *El modelo Scrum*. [En línea] 2011. http://eva.uci.cu/file.php/161/Documentos/Materiales_complementarios/UD_1_Procesos/Metodologias/SCRUM.
- Parada Soto, Omar, Botello Romero, Ninfa y Arias González, Maira. 2005.** Diseño e Implementación de un Sistema de Vigilancia Tecnológica para la provincia Granma. [En línea] Abril de 2005. <http://grciencia.idict.cu/index.php/granmacien/article/download/113/339>.
- Pecos, Daniel. 2010.** PostGreSQL vs. MySQL. [En línea] 2010. [Citado el: 30 de octubre de 2013.] http://danielpecos.com/docs/mysql_postgres/x15.html.
- Pérez, Javier Eguíluz. 2009.** Introducción Javascript. [En línea] 2009. [Citado el: 30 de octubre de 2013.] <http://librosweb.es/javascript/>.
- Pressman. 2005.** *Ingeniería de Software. Un enfoque práctico. 5ta Edición*. 2005.
- Saéz Domingo, Daniel y Antolín Fernández, Maria. 2008.** La Vigilancia Tecnológica aplicada al sector de las TIC. [En línea] 2008. http://www.iti.es/media/about/docs/tic/15/Articulo_2_15.pdf.

- Sparx Systems Pty. Ltd. 2007.** *Guía de Usuario de Enterprise Architect*. [En línea] 2007. <http://www.sparxsystems.com.ar/download/ayuda/index.html?deploymentdiagram.htm>.
- Tedeschi, Nicolas. 2014.** ¿Qué es un Patrón de Diseño? *Microsoft Network Developer*. [En línea] 2014. [Citado el: 28 de febrero de 2014.] <http://msdn.microsoft.com/es-es/library/bb972240.aspx>.
- Vega Prieto, Roexcy, Rodríguez Luis , Zayli y Justo Morell, Yaneisi Ofelia. 2010.** Procedimiento para realizar pruebas de usabilidad. [En línea] 2010. http://www.informatica-juridica.com/trabajos/procedimiento_para_realizar_pruebas_de_usabilidad.pdf.
- W3C. 2008.** Guia Breve de CSS. [En línea] 2008. [Citado el: 29 de octubre de 2013.] <http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo>.
- Wells, Don. 2009.** Extreme Programming: A gentle introduction. [En línea] 2009. [Citado el: 30 de octubre de 2013.] <http://www.extremeprogramming.org/>.

GLOSARIO

CMS: Sistema Gestor de Contenidos. Es un software para el manejo de contenidos de sitios Web directamente desde el servidor.

CSS: (Hoja de Estilo en Cascada). Se utiliza para definir el estilo dentro de las plantillas de diseño.

Hooks: Del inglés gancho, palabra utilizada en la comunidad de Drupal para nombrar los métodos de las interfaces que deben implementar los módulos desarrollados para este CMS.

Módulos: son ficheros de código que permiten ampliar las capacidades básicas de Drupal.

PHP: *Hypertext Preprocessor*, es un lenguaje de secuencia de comandos de servidor diseñado especialmente para el desarrollo de páginas Web dinámicas.

SGBD: Sistema Gestor de Base de Datos, en inglés *DataBase Management System* (DBMS), es una aplicación que permite a los usuarios definir, crear y mantener bases de datos, proporcionando un acceso controlado a las mismas.

HTTPS: *Hypertext Transfer Protocol Secure* (en español: Protocolo seguro de transferencia de hipertexto), es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de Hipertexto, es decir, es la versión segura de HTTP.

HTTP: *Hypertext Transfer Protocol* (en español: Protocolo de Transferencia de Hipertexto), es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre una PC cliente y un servidor Web.