UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 3

TÍTULO: Plataforma para la construcción de sistemas de digitalización de documentos (DigiPRO)

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

AUTORES: Fidel Alejandro Cebrián La Rosa

Jesús Soto Mitjans

TUTOR: Ing. Yadira Lizama Mué

CO-TUTOR: Ing. Yanet Edghill Martínez

Ciudad de La Habana, Junio de 2014

"Año del 55 Aniversario del Triunfo de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores del presente trabajo y reconocemos a la Facultad 3 de la Universidad de las Ciencias Informáticas; así como al Centro de Gobierno Electrónico los derechos patrimoniales del mismo, con carácter exclusivo.

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

_____

Jesús Soto Mitjans

Firma del autor

_____

Fidel Alejandro Cebrián La Rosa

Firma del autor

_____

Ing. Yadira Lizama Mué

Firma del tutor

AGRADECIMIENTOS

Jesús:

Son muchas las personas especiales a las que me gustaría agradecer su amistad, apoyo, ánimo y compañía en las diferentes etapas de mi vida. Algunas se encuentran presentes en este día y otras en mis recuerdos y en el corazón. Sin importar en donde estén o si alguna vez llegan a leer estas palabras quiero darles las gracias por formar parte de mí y por todo lo que me han brindado. Primeramente agradecerles a todas las personas que se encuentran aquí presentes para brindarme su apoyo. Agradecer al claustro de profesores que tuve durante estos años y que ayudaron de una forma u otra a mi formación profesional. A mis compañeros de aula por soportarme todos estos años y con los cuales pasé hermosos momentos, en especial a 3 muchachas que me ayudaron mucho y sin las cuales no hubiera llegado hasta aquí: Aymee, Yamilka y Anidey. A mis compañeros de apartamento Michel y Joel, realmente fue un placer haberlos conocido y compartir con ustedes todos estos años. A mis amigos del barrio Alejandro, José Antonio, Morgado, Marcel, Cynthia, Javier y René, gracias por su amistad todos estos años y los momentos inolvidables que juntos hemos compartido. A mi amiga Giselle por brindarme su ayuda cuando más necesitaba de ella y demostrar que la verdadera amistad trasciende a través de la distancia y el tiempo. A los profesores Dailien y Eliober por todos los consejos dados en aras de mejorar el presente trabajo. A nuestra co-tutora Yanet y al profesor Daniel por el tiempo dedicado y las ideas que aportaron a este trabajo. A mi tutora Yadira, por su dedicación y labor durante todo este trabajo, por sus inconformidades educativas con vista a obtener un trabajo con la mejor calidad posible y por verter toda su experiencia y empeño en la realización de esta tesis. A mi amigo y compañero de tesis Fidel, que sin él este trabajo no hubiera sido posible. A toda mi familia por su apoyo incondicional durante todos estos años. A mi tío Frank por estar siempre pendiente a mí. A mi abuelo Martín que no se encuentra entre nosotros, pero donde quiera que estés, siéntete orgulloso de tu nieto. A mi abuela Iraida por ser siempre positiva, al final tenías razón, todo salió bien. A mi hermana Beatriz por ser el motor que me impulsa a ser mejor cada día y poder servirte de ejemplo. A mis padres por ser las personas más importantes de mi vida, por ser mis guías, mis compañeros y mis amigos. Por apoyarme en todo momento, por impulsarme en los momentos más difíciles de mi carrera y por inspirarme a intentar ser cada día una mejor persona. Por ser ejemplos de sacrificio y entrega durante todos estos años de vida. Todo lo que soy hoy, es gracias ustedes. A todas aquellas personas que de una forma u otra ayudaron a la realización de este trabajo de diploma.

AGRADECIMIENTOS

Fidel:
Primeramente le quiero dar las gracias a esas dos personas a quienes le debo todo lo que tengo y todo lo que soy, a ellos que han sufrido conmigo cada golpe y gozado cada triunfo, a las personas con las que siempre he podido contar y han puesto mis metas justo al alcance de mis manos, dejándome a mí, solamente tomarlas, a mis padres, quienes espero que hoy estén más felices y orgullos que nunca de mí. También quiero agradecer a la profesora Yadira, por todo lo que nos ha ayudado durante estos dos últimos años de carrera especialmente durante el desarrollo del presente trabajo, aguantando con nosotros prácticamente hasta el nacimiento del pequeño Fidel Alberto, a quien quiero pedir disculpas por el tiempo de alimentación y cariño que le robamos. Profe, como no existe forma de expresarle el aprecio y admiración que tanto mi compañero de tesis como yo le tenemos, solo me resta decirle, gracias. Quiero agradecer también a nuestra co-tutora Yanet así como al profesor Daniel Varona, por sacrificar tantas noches de descanso, por su empeño e inconformidad ante lo mal hecho, así como por tomarnos bajo su tutela cuando lo necesitamos, por defendernos y apoyarnos sin descanso y mantenernos siempre en alto nuestro espíritu y nuestra confianza. A los profesores Dailien y Eliober por sus irremplazables consejos los cuales también fueron puntos fuertes en la realización de esta tesis. También quiero agradecer desde mi corazón a mi segunda y más reciente familia, Rosalía, José, Raiza y Yunet, por brindarme su hogar, sus conocimientos, su apoyo y más importante, sus corazones. De igual manera le doy las gracias a la niña de mis ojos, Thalía, quien ha estado a mi lado en este último año haciendo que cada minuto la ame un poquito más, gracias por ser como eres, por brindarme tu amor, tu comprensión, por permitirme entrar a tu familia y adueñarme de tu corazón. Quiero agradecer de forma especial, a alguien a quien considero más que un amigo un hermano, Alberto, gracias por soportarme por estos 5 años, por aconsejarme, por cuidarme, por ser mi mano derecha para todo, por ser mi confidente, por ser incondicional, por no dejarme solo en mis momentos más difíciles, por crear junto a mí las mayores chiquilladas que ha conocido esta universidad y ganarnos el título de PAYASOS de la facultad 3. También quiero agradecer a mis escuderas incondicionales Yamilka, Yordanka y más recientemente Dania, quienes junto a Alberto hemos formado el equipo élite de la facultad, el TEAM TRIGO, gracias por estar a mi lado y aguantarme mis pesadeces y momentos difíciles, por escucharme y aconsejarme, las quiero. Agradezco también a mi amigo y compañero de tesis Jesús por sus interminables obstines, y por mantenerse incansablemente en la lucha por la perfección de nuestro trabajo. Agradezco a mis amigos Charlie, Dian, Michel, Joel y Gabriela por su apoyo y disposición cuando los necesitaba. Agradezco a cada uno de los miembros del tribunal quienes contribuyeron a lograr una mayor correctitud en el presente trabajo así como a cada uno de los profesores quienes fueron parte esencial en mi formación profesional. Por último y no menos importante a cada una de las personas que me acompañan en este tan importante día solo me resta decirles, gracias por estar aquí, espero que sepan todos que aquí tienen un amigo con el que pueden contar el día y en el momento que lo necesiten. Gracias.

DEDICATORIA

Jesús:
Dedico este trabajo a mis padres que desde pequeño siempre me han guiado por el buen camino y han confiado siempre en mí, por todo ese amor y cariño que me han brindado, a ustedes, que hicieron que este sueño se hiciera realidad.

Fidel:
Le dedico este trabajo de diploma a las personas más importantes en mi vida, mis padres, quienes en cada segundo de cada hora de cada día, han estado velando por mi tranquilidad y bienestar para poder realizar este sueño, también le dedico este trabajo a otra persona que aunque no está presente hoy aquí, siempre ha velado por mí, mi abuela Luisa.

RESUMEN

Como parte del proceso de informatización de los principales sectores del país tanto en la esfera económica, como en la política y la social, en el Centro de Gobierno Electrónico (CEGEL) de la Universidad de las Ciencias Informáticas (UCI), se ha concebido el proyecto DigiPRO, el cual tiene como objetivo desarrollar una plataforma que permita agilizar la construcción de sistemas para la obtención de objetos digitales con valor legal (*). Para consolidar una idea sobre el modo en que se debe desarrollar la plataforma se llevaron a cabo métodos de investigación teóricos que permiten estudiar cómo ha sido el desarrollo de sistemas para la obtención de objetos digitales con valor legal, realizando un análisis detallado de cada uno de los mismos. Con respecto a la metodología de desarrollo utilizada se establece el uso de la metodología Rational Unified Process para el desarrollo de las etapas del proyecto, aunque la elaboración de los artefactos del Expediente de Proyecto está regida bajo las normas establecidas por el Programa de Mejora CMMI Nivel 2 que enfrenta la UCI actualmente. El desarrollo de la plataforma ha logrado la reducción del tiempo de desarrollo de cada uno de los sistemas implementados usando la misma, lo que representa una posibilidad de aumentar la explotación de este mercado con que cuenta el Centro actualmente.

Palabras claves: Digitalización, Plataforma

(*) Cualquier fichero digital que haya sido obtenido a partir de un proceso legal como inscripciones de nacimiento y sentencias digitales de juicios.

ÍNDICE DE CONTENIDOS

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE FIGURAS

Introducción

La digitalización de documentos es una tecnología factible reconocida desde hace varios años en función de la conservación y preservación de documentos (1) (2). Se entiende por digitalizar a los intereses de esta investigación, como el proceso de obtención de un objeto digital con valor legal a partir de un documento en formato duro, teniendo en cuenta las ventajas que ofrece este tipo de actividad como son la gestión inmediata de la información disponible, acceso simultáneo a la información y manipulación escasa de la documentación física. Esto ha provocado que muchas instituciones hayan optado por el desarrollo de este tipo de tecnología en función de alcanzar agilidad, sustentabilidad y modernización en los procesos que desarrollan, así como minimizar los altos costos en materia de preservación que implica la constante manipulación de su patrimonio documental (3) (4) (5). Por otro lado el auge de grandes empresas dedicadas al desarrollo de esta tecnología (6) (7) (8) muestra resultados significativos en esta área, por lo que las pequeñas y medianas empresas deben buscar alternativas viables y especializadas para avanzar en este mercado atendiendo a las necesidades de su sector más cercano.

En Cuba, el Centro de Gobierno Electrónico, perteneciente a la Facultad 3 de la Universidad de las Ciencias Informáticas, tiene como misión satisfacer necesidades de clientes gubernamentales mediante el desarrollo de productos, servicios y soluciones integrales de alta confiabilidad, calidad, competitividad, fidelidad y eficiencia a partir de un personal altamente calificado. Entre las líneas de investigación que desarrolla se encuentran la Informática Jurídica Documental y la Protección del Patrimonio Digital; ambas están estrechamente relacionadas a los procesos de generación, gestión y preservación de objetos digitales con valor legal. Desde su creación, el Centro se ha destacado en la comercialización de productos para la digitalización de documentos que han aportado resultados considerables a la Universidad y a la economía del país. Estas soluciones son: Centro de Digitalización para el Servicio Autónomo de Registros y Notarías Públicas, Centro de Digitalización para la División de Antecedentes Penales y Centro de Digitalización de Alfabéticas para el Servicio Administrativo de Identificación, Migración y Extranjería; actualmente están algunas de las mismas desplegadas en la República Bolivariana de Venezuela con resultados satisfactorios. Aunque este tipo de sistemas tienen un conjunto de características similares respecto a los procesos de negocio que informatizan, en la práctica se ha observado que su desarrollo aislado ha propiciado un conjunto de deficiencias que fundamentan la problemática como son:

- Escaso nivel de configuración de los componentes de las soluciones, lo que limita las posibilidades de adaptación a otras tipologías de fondos documentales con valor legal.
- Escasas posibilidades de reutilización de software.
- Variedad en la plataforma tecnológica de desarrollo, en la mayoría de los casos utilizando tecnologías costosas que pueden ser remplazadas por tecnologías libres.
- Falta de estandarización de mecanismos de resguardo, migración de datos y preservación de los objetos digitales que garantice la confiabilidad, integridad y disponibilidad de la información digital para la gestión de la misma.
- Falta de estandarización de los mecanismos para la internacionalización de la aplicación, así como para el uso de certificados digitales y políticas de administración y configuración que atentan contra la consolidación de un sello único de productos.
- Falta de estandarización de los mecanismos para la incorporación de módulos y funcionalidades, así como para la presentación del marco de trabajo y apariencia.

Estos elementos traen como consecuencia el retraso en los tiempos de desarrollo de sistemas similares para nuevas entidades, lo que afecta las posibilidades de explotación de este mercado que actualmente tiene el Centro, principalmente las relacionadas a la Internacionalización, la presentación de la apariencia y la incorporación de módulos, siendo estas últimas las principales a erradicar con el desarrollo de la aplicación.

Ante esta situación se propicia la creación del proyecto Plataforma para la construcción de sistemas de digitalización de documentos (DigiPRO) (9). Este persigue el objetivo de desarrollar una solución tecnológica que permita agilizar la construcción de sistemas para la generación de objetos digitales con valor legal con la automatización de los procesos de un Centro de Digitalización, propiciando la disminución del tiempo de desarrollo y comercialización de los mismos, la modernización e informatización de la gestión de objetos digitales con valor legal, así como la preservación y protección de fondos documentales. Este objetivo se desagrega, como lo contempla su proyecto técnico, en las siguientes pautas a cumplir:

- Desarrollo de una plataforma para la construcción de sistemas de digitalización de documentos que incluye las etapas de desarrollo de software: Modelamiento del Negocio, Requisitos, Análisis y Diseño, Implementación, Prueba.
- Garantizar la generación de objetos digitales con valor legal favoreciendo la gestión de dichos elementos en función de la calidad, confiabilidad, autenticidad e integridad del patrimonio

documental digital.

- Concepción y refinamiento de los procesos de un Centro de Digitalización para la generación de objetos digitales con valor legal que contemple la especificación de áreas, procesos, recursos, roles y actividades de forma que exista flexibilidad para la adaptación a entornos con exigencias de negocio diferentes.

Referente a la primera pauta de desarrollo se concibe el subsistema Plataforma como parte de la solución tecnológica cuyo objetivo principal es permitir la creación y configuración de sistemas para la generación de objetos digitales con valor legal a partir de un conjunto de componentes básicos previamente incorporados de manera que permita agilizar el proceso de creación de sistemas para la informatización de centros de digitalización flexibles a las exigencias del negocio de los diferentes entornos institucionales.

Es importante destacar que estos componentes previamente incorporados están contemplados en el subsistema Base del proyecto y su lógica de negocio está en función de los procesos de un Centro de Digitalización flexible a diferentes tipos de fondos documentales.

En el año 2012 se completó la fase de Inicio del proyecto teniendo como hitos fundamentales el estudio preliminar con la planificación de las etapas y el estudio de factibilidad de su ejecución. Posteriormente la fase de Elaboración se desarrolló hasta obtener como resultado importante la especificación de requisitos de software aceptada y firmada por el Centro después de un proceso de revisión por la empresa cubana CALISOFT. Partiendo de la problemática anteriormente descrita se plantea el siguiente **problema a resolver:** ¿Cómo satisfacer los requisitos acordados con el Centro CEGEL del subsistema Plataforma para estandarizar la construcción de sistemas para la generación de objetos digitales con valor legal?

Se plantea como **objeto de estudio:** Procesos de creación y configuración de aplicaciones.

Para dar solución al problema anteriormente planteado se ha propuesto el siguiente **objetivo general:** Diseñar e implementar el subsistema Plataforma del proyecto DigiPRO de manera que se logren satisfacer los requisitos acordados con la institución para la creación y configuración de sistemas para la generación de objetos digitales con valor legal.

El **campo de acción** es: Plataformas para la construcción de sistemas para la generación de objetos digitales con valor legal.

Para el cumplimiento del objetivo general se plantean los siguientes **objetivos específicos:**

- Elaborar el marco teórico de la investigación.
- Diseñar la propuesta de solución.

- Implementar los elementos del diseño obtenidos.
- Validar la solución propuesta.

Teniendo como **idea a defender:** Con el diseño e implementación del subsistema Plataforma del proyecto DigiPRO, se logrará estandarizar la construcción de sistemas para la generación de objetos digitales con valor legal, de manera, que se logren satisfacer los requisitos acordados con la institución para la creación y configuración de sistemas para la generación de objetos digitales con valor legal.

Para cumplir los objetivos antes propuestos se han diseñado las siguientes **tareas de investigación y desarrollo:**

- Caracterización de los procesos de creación y configuración de sistemas para la obtención de objetos digitales.
- Caracterización de soluciones para la obtención de objetos digitales con valor legal.
- Caracterización de las tecnologías y herramientas seleccionadas para el desarrollo de la propuesta de solución.
- Análisis de los requisitos funcionales y no funcionales del subsistema Plataforma.
- Descripción de la arquitectura del sistema.
- Diseño de la solución propuesta en función de los requisitos especificados.
- Implementación de los elementos de diseño obtenidos.
- Validación de la implementación del sistema a partir de la aplicación de pruebas unitarias y funcionales.
- Validación de la solución propuesta a partir de la aplicación de métricas para el tiempo de desarrollo.

Se utilizan durante el desarrollo de la propuesta los siguientes **métodos de investigación**:

Métodos teóricos:

- ✓ **Histórico – Lógico:** permite el análisis del comportamiento del desarrollo de sistemas para la obtención de objetos digitales con valor legal en las últimas décadas, permitiendo obtener conclusiones lógicas sobre los elementos que distinguen a este tipo de software, la necesidad de introducir la plataforma para la agilización y estandarización de su construcción y las características que se han consolidado en el tiempo y que deben estar presentes en la propuesta de solución identificada.
- ✓ **Análisis y Síntesis:** permite el análisis de la bibliografía consultada referente a la investigación, fundamentalmente los conceptos relacionados al proceso de creación y configuración de

aplicaciones, los sistemas para la obtención de objetos digitales con valor legal y las herramientas a utilizar para el desarrollo de la propuesta; así como sintetizar los elementos más relevantes afines al presente trabajo.

- ✓ **Modelación:** permite crear una abstracción de la realidad de la propuesta de solución antes de llevar a cabo su implementación. Es utilizado para la obtención de los elementos del diseño a partir de la especificación de requisitos funcionales y no funcionales del subsistema Plataforma.

El presente trabajo se estructura en 3 capítulos que se describen a continuación:

Capítulo 1 Fundamentación teórica: este capítulo muestra el análisis de los principales elementos del marco teórico de la investigación introduciendo los conceptos relacionados a los procesos de creación y configuración de aplicaciones, presentando una caracterización de los sistemas para la obtención de objetos digitales con valor legal que sirven de referencia tanto en Cuba como en el mundo; así como la fundamentación de la selección de las herramientas seleccionadas para el desarrollo de la propuesta y la metodología que guiará el proceso.

Capítulo 2 Características y diseño del sistema: en este capítulo se presentan las características del subsistema Plataforma, los requisitos funcionales y no funcionales que debe satisfacer la solución y la especificación del sistema en función de los casos de uso del análisis. Se muestra además, la descripción de la arquitectura que soporta la plataforma, los principales resultados del diseño expresados en los diagramas y la descripción de las clases, así como la validación del trabajo realizado en esta etapa a partir de métricas de diseño de clases.

Capítulo 3 Implementación y prueba: en este capítulo se describe el proceso de implementación de la propuesta presentando los elementos del modelo de despliegue y diagramas de componentes. Además se muestran los resultados del análisis del proceso de verificación y validación de la solución con la aplicación de pruebas de software y la validación de la idea a defender presentada en la investigación.

CAPÍTULO 1 Fundamentación teórica

1.1 Introducción

El presente capítulo aborda los resultados del análisis de los principales elementos del marco teórico de la investigación introduciendo los conceptos relacionados a los procesos de creación y configuración de aplicaciones, presenta una caracterización de los sistemas para la obtención de objetos digitales con valor legal que sirven de referencia a la investigación tanto en Cuba como en el mundo; y fundamenta la selección de las herramientas para el desarrollo de la propuesta y la metodología que guiará el proceso.

1.2 Introducción a los procesos de creación y configuración de aplicaciones

Según Jacobson, Booch y Rumbaugh *"en la ingeniería de software el objetivo es construir un producto de software o mejorar uno existente*" (10). Opinión que convierte al producto en el centro del proceso de desarrollo, por lo que el propio proceso de creación en sí es, desde el punto de vista metodológico, lo más importante.

Las metodologías de desarrollo de software, especialmente las clásicas o pesadas, definen un conjunto de fases y flujos de trabajo contenidas en el proceso de desarrollo y por las cuales debe pasar el producto durante todo el ciclo de vida. En el caso de RUP (Rational Unified Process) por ejemplo, el proceso abarca 4 fases de desarrollo (Inicio, Elaboración, Construcción y Transición) y 9 flujos de trabajo (Modelado de Negocio, Requisitos, Análisis y Diseño, Implementación, Pruebas, Despliegue, Configuración y Control de Cambios, Gestión de Proyecto y Ambiente).

La configuración es un elemento muy importante también porque permite la flexibilidad del sistema ante determinados cambios ya previstos por los involucrados. Mientras más opciones de configuración se diseñen e implementen correctamente se considerará que el sistema será más adaptable a las necesidades de los clientes. Las opciones de configuración deben definirse desde la Especificación de requisitos, ser consideradas como requisitos propios del sistema y están estrechamente relacionadas a las posibilidades de cambio del resto de los requisitos. Es por eso que su definición involucra tanto a desarrolladores como clientes.

Existen dos momentos en los que puede configurarse un sistema:

- **A partir de la etapa de despliegue:** En este caso las opciones fueron diseñadas e implementadas para que el responsable de la configuración sea el propio cliente. El sistema, ya en explotación, se adapta en función de la configuración establecida sin necesidad de hacer cambios en el código

fuente, recompilar y reinstalar. Generalmente las opciones en este caso no son muy amplias porque cada una de ellas representa una o varias bifurcaciones que tuvo que tenerse en cuenta durante la implementación de los requisitos, lo que lo hace más engorroso. Sin embargo es el escenario más utilizado en la actualidad.

- **Durante el propio proceso de creación:** En este caso se demanda un proceso de desarrollo mejor automatizado. Las actividades de análisis, diseño e implementación del producto demandan menor esfuerzo porque existen componentes previamente diseñados con un grado de configuración tal que es posible ensamblarlos en un producto para satisfacer los requisitos de software solo modificando sus opciones de configuración. Este escenario puede ser recreado cuando el negocio es bien descrito. Los sistemas de gestión de contenido son un buen ejemplo.

La necesidad de automatizar los procesos de creación y configuración de productos de software es constante y creciente a medida que estos alcanzan mayor nivel de complejidad. Esta necesidad ha estado influenciada por diferentes factores como:

- La insistencia de los clientes por acortar los períodos de desarrollo de software porque necesitan el producto lo antes posible. Este fenómeno es bastante común en la actualidad y exige agilidad en la construcción de aplicaciones para satisfacer las expectativas y ser competitivos en el mercado.
- El auge del desarrollo basado en componentes (11) que amplía las posibilidades de reutilización (12) (13) (14) (11) y fortalece en gran medida el proceso de ingeniería, ha propiciado un entorno favorable para que la creación de productos de software parta desde la reutilización y ensamblaje de componentes de software (15).
- La necesidad de propiciar un entorno favorable para la estandarización de procesos, refinamiento de componentes, transparencia en el mantenimiento, políticas de usabilidad como la gestión de idiomas, la apariencia, la presentación de interfaces consistentes, requisitos comunes de administración y configuración, entre otros que garantizan la creación de un sello único de productos.

Dentro del proceso de desarrollo de software las herramientas, entendidas como “el software que se utiliza para automatizar las actividades definidas en el proceso” (10) juegan un papel importante. Se emplean durante todo el ciclo de vida del software fundamentalmente en el proceso de creación y configuración del producto en sí. En este caso se destacan los Entornos Integrados de Desarrollo (IDE, por sus siglas en inglés): programas informáticos compuestos por un conjunto de herramientas que facilitan la obtención del código fuente; las Herramientas de Ingeniería de Software Asistida por Computadoras (CASE, por sus siglas

en inglés): destinadas fundamentalmente a la modelación, el diseño, cálculo de costos, generación de código automático y documentación del producto (14); los Marcos de Trabajo o *frameworks*, que tienen una estructura conceptual y tecnológica de soporte definido y que sirven de base para el desarrollo de las aplicaciones; y las Plataformas de desarrollo que son más amplias abarcando elementos de hardware y/o software.

1.3 Plataformas para el desarrollo de aplicaciones

En informática, una plataforma es un sistema que sirve como base para hacer funcionar determinados módulos de hardware o de software con los que es compatible. Dicho sistema está definido por un estándar alrededor del cual se determina una arquitectura de hardware o una plataforma de software (incluyendo entornos de aplicaciones) (16) .

En la actualidad esta definición origina una amplia diversidad en la tipología. Unas están compuestas en mayor grado de elementos de hardware (17) (18) (19) (20), otras por software (21) (22) y otras por una combinación de ambos (23). Con el auge de la Cloud Computing (24) se ha expandido la creación de plataformas de hardware y/o software para la prestación de servicios.

Las plataformas destinadas al desarrollo de aplicaciones informáticas traen consigo diversas ventajas como son la **agilidad en el desarrollo** ya que las tareas que por lo general toman largos períodos de tiempo se pueden hacer en cuestión de minutos con el uso de funciones pre-construidas. El desarrollo se torna más fácil, más rápido y por lo tanto **eficaz**. Otra ventaja es la **seguridad**, una plataforma ampliamente utilizada tiene mejores implementaciones de seguridad, porque los usuarios finales se convierten en probadores a largo plazo y las actividades de mantenimiento, detección de errores y bugs se facilitan y robustecen. En su mayoría vienen respaldadas por una documentación y una amplia comunidad que a través de foros brindan una rápida respuesta a cualquier inquietud creando facilidades para la realización de las actividades de **soporte**.

Una vez identificada la necesidad del uso de una plataforma para la creación de productos de software debido a las ventajas que ofrece y su adecuación al escenario presentado, es necesario el estudio de experiencias previas en la digitalización de documentos para identificar características de referencia que pueden ser automatizadas durante el proceso de creación de estos sistemas.

1.4 Sistemas para la digitalización de documentos: Características de referencia

Para determinar las características de referencia de sistemas para la digitalización se presenta el estudio de varios sistemas y modelos desarrollados con estos fines y que tienen un alcance reconocido en Cuba y

el mundo. No todos son especializados en la obtención de objetos digitales con valor legal pero se consideran otros aportes desde el punto de vista funcional.

En el mundo

- **DIGIBÍS:** Empresa española especializada en digitalización, creación de Bibliotecas Virtuales o Digitales en Internet y otros soportes electrónicos y desarrollos propios de programas de gestión digital para bibliotecas, archivos y museos que se actualizan de acuerdo con la normativa internacional para el intercambio de información. Su principal objetivo es promover la máxima visibilidad en la WWW y favorecer el intercambio de información entre las webs españolas, en particular las bibliotecas, archivos y museos digitales de entidades públicas y privadas, y la comunidad internacional. Tiene la misión de ayudar a preservar y difundir el Patrimonio Cultural español (25).

  Su plataforma está integrada por varios sistemas de software especializados donde destaca para nuestra investigación debido a su función:

  - **Sistema Digital de Gestión de Archivos (DIGIARCH):** herramienta que permite la descripción de fondos archivísticos en un entorno estableciendo una relación jerárquica entre los diferentes elementos descritos: fondo, secciones, series y expedientes (26).

  Tanto este como el resto de los sistemas desarrollados por DIGIBÍS son soluciones específicas a problemas puntuales y son sistemas privativos, lo que limita el uso de los mismos por otras entidades.
- **Microfile:** empresa líder en digitalización de documentos de Uruguay. Esta cuenta con moderna tecnología y personal capacitado para digitalizar y procesar volúmenes de documentos considerables. Ofrece, a través de sus servicios, la posibilidad de disponer en formato digital de toda la documentación. El servicio comprende (27):
  - **Retiro de la documentación:** disposición de servicios para efectuar el retiro de los documentos en los puntos que sean solicitados y con la frecuencia deseada.
  - **Preparación documental:** eliminación de grapas, cintas, sobres y cualquier otro tipo de elemento como post-its, realizándose también en caso de requerirlo, una clasificación del material.
  - **Digitalización:** obtención de la imagen digital del documento físico. En este proceso se realizan controles de calidad a la imagen.

- **Indexación:** los documentos se indexan o catalogan para luego poder ser localizados por el usuario. Esta puede ser automática (OCR, Códigos de barras, etc.) o manual.
- **Control de calidad:** se utilizan diferentes métodos para garantizar la calidad y la integridad de los datos.
- **Integración de los documentos:** los documentos pueden ser gestionados a través de un software de Administración Electrónica de Documentos como FSEDM, o incorporarlos a los nuevos sistemas, etc.

El alcance de esta aplicación es reconocido en su entorno, sin embargo los procesos que intervienen en la digitalización se consideran insuficientes para la obtención de objetos digitales con valor legal al igual que el proceso de indexación para la obtención de metadatos. Además tiene restricciones comerciales para su uso.

- **XEROX-DocuShare:** Xerox Corporation es una empresa global líder en procesos comerciales y gestión de documentos. Entre las distintas herramientas que posee esta empresa se encuentra **DocuShare** utilizada para la digitalización de documentos. Esta aplicación almacena digitalmente archivos que antes se conservaban en papel. Es una solución novedosa de fácil implementación que permite, entre otras cosas, visualizar cada documento, clasificarlo de acuerdo a diversos parámetros y extraer informes estadísticos. Funciona de la siguiente manera: una vez que los documentos están listos para ser archivados, son enviados a un laboratorio de Xerox, donde se digitalizan a través de un proceso de escaneo que automáticamente mejora la calidad de la imagen e indexa la información de acuerdo a parámetros como fecha, código de materia, nombre y apellido de alumno, código de matrícula, código de la carrera, y calificación. Posteriormente la información se cifra mediante un código de certificación MD5 que permite corroborar la autenticidad de la información contenida (28) .

  A pesar de su amplia difusión y aplicación para la obtención de objetos digitales, esta herramienta no garantiza el valor legal de los mismos debido a la no especialización de fondos documentales que poseen. Tiene restricciones comerciales de uso y está protegida por licencias de software.

Los anteriormente mencionados sistemas tienen una aproximación amplia a la solución deseada pero no pueden ser parte de la propuesta porque son sistemas costosos y no garantizan la obtención de objetos digitales con valor legal.

En Cuba

En Cuba son reconocidos los resultados alcanzados por el Centro de Gobierno Electrónico en el desarrollo de sistemas para la digitalización de fondos documentales con valor legal.

DigiPyrus

Ante la necesidad del Ministerio del Poder Popular para las Relaciones del Interior y Justicia (MPPRIJ) de la República Bolivariana de Venezuela de digitalizar los folios correspondientes al asiento registral de un grupo inicial de oficinas del Servicio Autónomo de Registros y Notarías Públicas se desarrolla este sistema como parte de la solución tecnológica integral para la automatización y modernización de estas oficinas (29).

Procesos

Los procesos que se informatizan (29) se muestran en la Figura 1 y se describen a continuación:

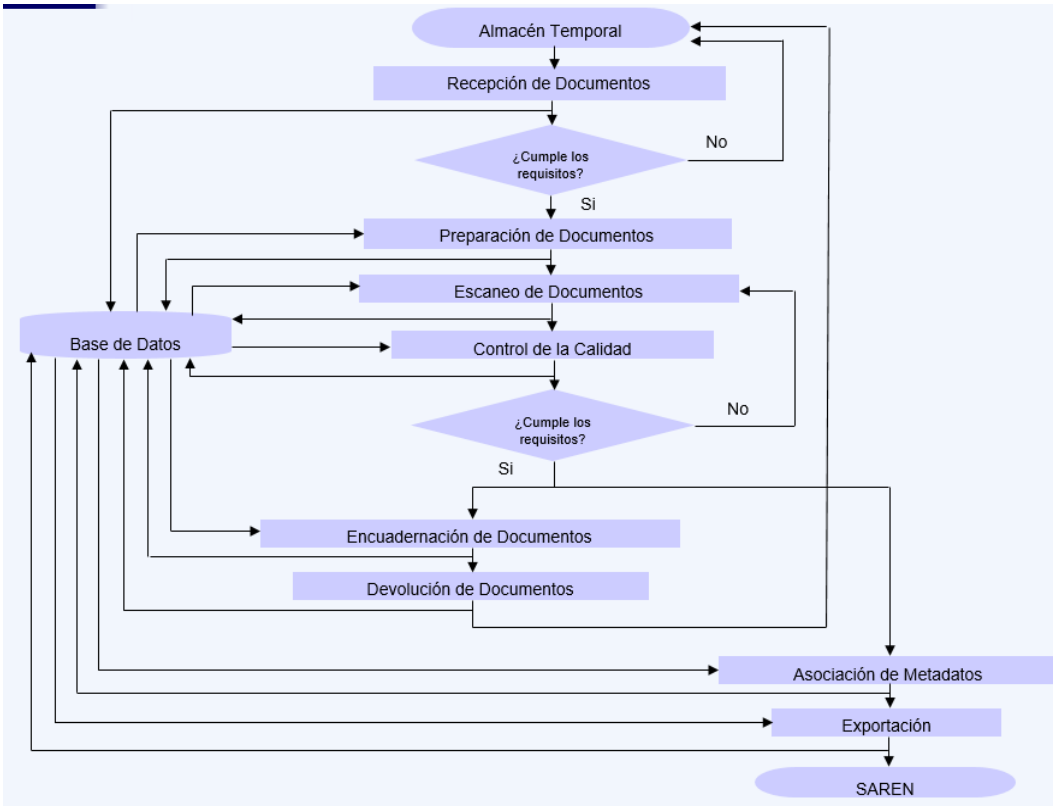

Figura 1 Flujo de procesos que informatiza DigiPyrus.

Recepción de Documentos: verificación y registro de los tomos provenientes del Almacén Temporal. Las operaciones principales que se realizan son:

- Registro del código del Transportista en la aplicación.
- Verificación de la cantidad de tomos en el Modelo de Recepción.
- Creación y registro del Lote de Entrada.

Preparación de documentos: ejecución de las operaciones establecidas para garantizar que los tomos cumplan con los parámetros necesarios para el proceso de digitalización. Las operaciones principales que se realizan son:

- Registro del tomo en el área.
- Desencuadernación de los tomos.
- Guillotinado del lomo del tomo.
- Separación de las páginas del tomo.
- Registro de la salida del tomo.

Escaneo de Documentos: conversión de los tomos físicos en objetos digitales multipáginas. Las operaciones principales que se realizan son:

- Registro del tomo en el área.
- Configuración del escáner.
- Escaneo de las páginas del tomo.
- Salva del objeto digital multipáginas.

Control de la Calidad: inspeccionar el 100% del objeto digital multipáginas proveniente del área de Digitalización de Documentos. Las operaciones principales que se realizan son:

- Registro del tomo en el área.
- Ejecución de las operaciones de limpieza de la imagen y bordes del tomo.
- Inspección del tomo físico y el objeto digital multipáginas.
- Colocación de las páginas el tomo físico en su caja.

Encuadernación de Documentos: recuperación y preservación de las condiciones iniciales de los tomos digitalizados. Las operaciones principales que se realizan son:

- Registro del tomo en el área.
- Revisión continua de los folios del tomo y emparejado de los mismos.
- Cosido de las páginas del tomo.
- Pegado de las guardas al lomo del tomo.

- Pegado del primer refuerzo del tomo.
- Pegado de la cabezada al tomo.
- Pegado del segundo refuerzo del tomo.
- Pegado de la carátula al tomo.
- Registro de salida del tomo en el área.
- Secado del pegamento.

Devolución de Documentos: registro de salida de los tomos que han sido digitalizados en el proceso. Las operaciones principales que se realizan son:

- Creación del Lote de Salida e impresión del Acta de Devolución.
- Entrega del Lote de Salida al Almacén Temporal.

Asociación de Metadatos: identificación de los datos de los objetos digitales multipáginas para la asociación de sus metadatos. Las operaciones principales que se realizan son:

- Cargar la imagen del tomo.
- Identificación de los datos necesarios presentes en el Acta de Apertura y Cierre así como en las Unidades Documentales del tomo.
- Salvar los metadatos asociados al tomo.
- Control de la Calidad de los metadatos asociados al tomo.

Exportación: exportación de las imágenes digitales multipáginas de los documentos del tomo al Centro de Datos del SAREN. Las operaciones principales que se realizan son:

- Selección de la cantidad de tomos listos a exportar.
- Realizar la exportación de los documentos del tomo.

Tecnologías de desarrollo

Las tecnologías empleadas para la implementación de la solución fueron FrameWork .NET como plataforma de desarrollo, Leadtools para la interacción con dispositivos de escaneo, NUnit para la realización de pruebas al código y Crystal Reports para la elaboración de reportes (30).

Descripción de la Arquitectura

La arquitectura del sistema se distingue por su distribución en tres capas: Presentación, Lógica de Negocios y Acceso a Datos, como se muestra en la Figura 2 Vista de la arquitectura del sistema DigiPyrus. (30).

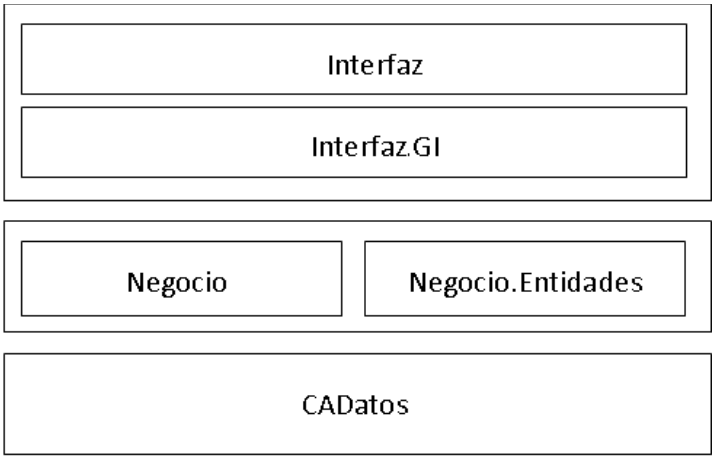

Figura 2 Vista de la arquitectura del sistema DigiPyrus.

DigiDAP

DigiDAP forma parte de la Solución Tecnológica Integral para la automatización y modernización de la División de Antecedentes Penales de la República Bolivariana de Venezuela, como el subsistema que informatiza los procesos del Centro de Digitalización para el fondo documental de dicha institución. Su objetivo fundamental es garantizar la obtención de objetos digitales con valor legal a partir de la digitalización del fondo para mejorar los servicios de inscripción y certificación de antecedentes penales a los ciudadanos de la nación venezolana. Este sistema tiene como limitante que es una solución específica ante una problemática determinada, lo que limita las posibilidades de reutilización de sus componentes ante otras oportunidades de negocio con diferentes fondos documentales.

Procesos

En el modelo de referencia para la digitalización de fondos documentales de antecedentes penales (31) (32) se establecen como procesos los que se describen a continuación en la Figura 3:

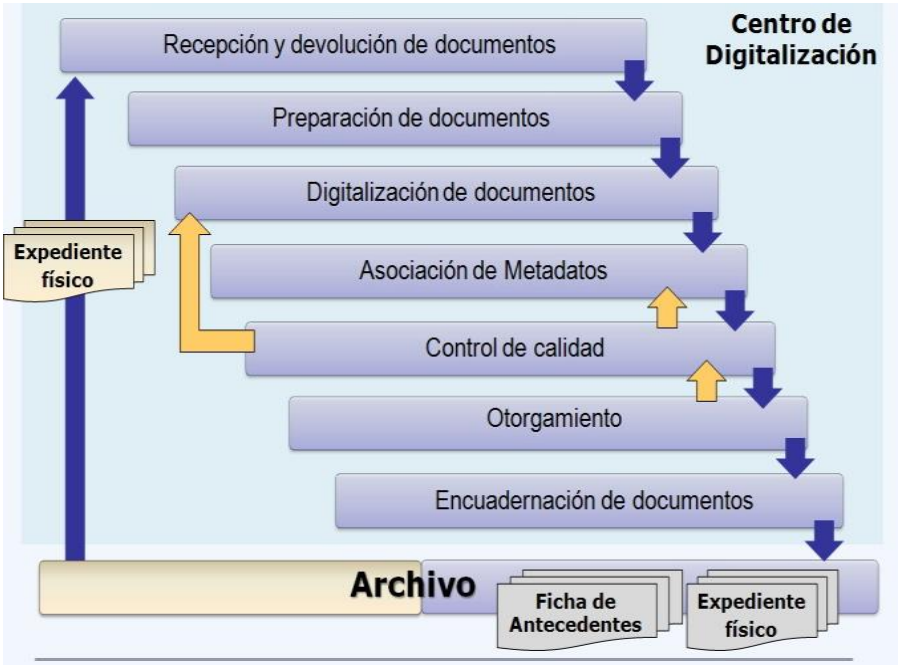

Figura 3 Flujo de procesos que informatiza DigiDAP.

Área de Recepción y Devolución de Documentos:

Funcionario de Recepción y Devolución de Documentos

1. Recepcionar expedientes al Centro de Digitalización.
2. Devolver Expedientes del Centro de Digitalización hacia el archivo.
3. Eliminar expedientes que por errores se decida retirar del proceso.

Área de Preparación de Documentos:

Funcionario Operador Gráfico

1. Preparar los expedientes y garantizar que tengan las condiciones necesarias para pasar por el escáner en la próxima área.
2. Evaluar la calidad de las unidades documentales para determinar si proceden o no en el proceso y registrar los folios a digitalizar de cada una.
3. Emitir notas de preparación en caso de que sea necesario para esclarecer la omisión de folios a digitalizar.

Área de Digitalización de Documentos:

Funcionario Operador de Digitalización

1. Digitalizar las unidades documentales contenidas en los expedientes.

2. Rectificar errores en los trámites que hayan sido regresados en el proceso.
3. Mejorar la calidad visual de los documentos digitales.

Área de Asociación de Metadatos:

Funcionario de Procesamiento de Datos

1. Asociar metadatos a las unidades documentales contenidas en los expedientes.
2. Identificar posibles coincidencias documentales para eliminar unidades documentales duplicadas en el archivo.
3. Identificar posibles asociaciones entre unidades documentales en expedientes.
4. Rectificar errores de metadatos en los trámites que hayan sido regresados en el proceso.

Área de Control de Calidad:

Funcionario de Control de Calidad

1. Revisar legalmente los trámites realizados en el Centro de Digitalización.
2. Emitir la respuesta correspondiente del trámite.
3. Imprimir salidas del sistema.

Área de Otorgamiento:

Director del Centro de Digitalización

1. Otorgar trámites revisados legalmente, certificando la calidad del proceso realizado.
2. Otorgar firma manuscrita a las salidas impresas del sistema y la firma digital a los documentos digitales.
3. Obtener reportes estadísticos sobre el proceso.
4. Localizar trámites dentro del proceso y consultar el estado de los mismos.

Área de Encuadernación:

Funcionario de Encuadernación de Documentos

1. Encuadernar y foliar las unidades documentales dentro de los expedientes para su asentamiento en archivo (33).

Tecnologías de desarrollo

Entre las tecnologías empleadas para su desarrollo (31) destacan Visual Paradigm como herramienta CASE para el modelado del sistema, UML como lenguaje de modelado, Java como lenguaje de programación, Swing como framework para la construcción de interfaces. EJB (Enterprise Java Beans) como framework para el desarrollo basado en componentes. Java Persistence API (JPA) como framework para la

persistencia de datos. PostgreSQL como sistema de gestión de bases de datos. GlassFish como servidor de aplicaciones.

Descripción de la Arquitectura

La arquitectura de este sistema tiene como principales características (32):

- Se distingue el estilo arquitectónico Cliente/Servidor

  En este caso, dicha arquitectura proporciona mejoras desde el punto de vista de la portabilidad de la aplicación, escalabilidad, robustez y reutilización de código con respecto a los sistemas anteriores, también facilita las tareas de migración o cambios en el sistema gestor de base de datos. Las principales ventajas son: uso de recursos centralizados debido a que el servidor es el centro de la red, puede administrar los recursos que son comunes a todos los usuarios, seguridad mejorada, administración al nivel del servidor y una arquitectura de red escalable.

  El sistema está compuesto principalmente por dos aplicaciones cliente y servidor, la cliente se instala en las estaciones de trabajo e interactúa con la otra desplegada en el servidor que proporciona los servicios adecuados para la realización de la lógica de negocio de la aplicación. La comunicación entre ambas se garantiza a través del estándar Enterprise JavaBeans (EJB).

- Distribuida en 4 capas

  La *capa de presentación* contiene los formularios con los que interactúan los usuarios finales del sistema y permite la visualización de toda la información que estos necesitan, así como las acciones que gestionan las funcionalidades de los formularios y garantizan la comunicación con los gestores de negocio en el cliente para la invocación de la lógica de negocio.

  La *capa de negocio cliente* contiene los gestores de negocio del cliente que utilizan las acciones, además la Fachada Gestores Remotos que sirve de fachada para la comunicación entre los gestores clientes y servidor.

  La *capa de negocio en el servidor* que contiene los gestores de negocio del servidor responsables de las tareas propias de la aplicación, implementan la lógica de la aplicación, aplican las reglas de negocio y las entradas del usuario.

  La *capa de acceso a datos* contiene los gestores de acceso a datos y las clases persistentes para la gestión de los datos almacenados en la capa de datos. Es responsable de la gestión y almacenamiento permanente de los datos. La capa de datos contiene los datos gestionados por la aplicación.

- Basada en el desarrollo de componentes.
  Los componentes son los gestores de negocio en el servidor, que son Beans de sesión remotos sin estado, los gestores de acceso a datos, que son Beans de sesión locales sin estado y las clases persistentes, que son Beans de entidades. Los Beans dirigidos por mensajes existen también para el paso de mensajes entre las aplicaciones cliente y servidor.

CDA

CDA es la solución tecnológica desarrollada para el Centro de Digitalización de Alfabéticas para el Servicio Administrativo de Identificación, Migración y Extranjería. Para su desarrollo se utilizó como marco de trabajo el utilizado por la solución DigiPyrus por lo que las características arquitectónicas y tecnologías de desarrollo utilizadas son similares.

Esta solución está compuesta por los siguientes módulos:

- **Almacén Temporal:** permite la gestión de inventario de los recursos que entran y salen del Centro de Digitalización, así como del control de recibo y despacho de las Tarjetas de Alfabética desde las Oficinas de Identificación hacia cada Centro de Digitalización.
- **Preparación:** permite la creación de los lotes de documentos en el sistema, se registran los documentos rechazados por malas condiciones para entrar en el proceso de digitalización y se realiza la corrección de los documentos rechazados por mala preparación o códigos de barra ilegible.
- **Digitalización:** permite la digitalización de los lotes de documentos o de documentos rechazados.
- Metadatos: permite la asociación de los datos a los documentos, se reportan los documentos rechazados por mala calidad de la imagen, se corrigen los documentos rechazados y se registran las pérdidas de documentos en caso de que existan.
- **Calidad:** permite llevar a cabo la revisión de la calidad de las imágenes digitales y los metadatos de los documentos, se reportan los documentos rechazados por mala calidad de la imagen o de los metadatos y se registran las pérdidas de documentos en caso de que existan.
- **Firma digital:** permite la emisión de la firma digital de los documentos, certificando que la información obtenida es verídica.
- **Reportes:** permite la gestión de información del centro mediante diferentes reportes que permiten conocer el estado de desempeño de los trabajadores y el estado de procesamiento de lotes, cajas y documentos dentro del Centro de Digitalización.

- **Gestión Documental:** permite el manejo de la gestión de información de los lotes y los documentos, lo cual posibilita conocer donde se encuentra un lote o documento en cualquier momento, liberar los documentos o los lotes que se encuentren en proceso por otro usuario y conocer que documentos fueron rechazados por el director para conocer su ubicación dentro del almacén.
- **Administración local:** permite la configuración interna del centro, la administración de usuarios, roles y contraseñas y el control de las sesiones de usuarios en el sistema.
- **Sesión:** permite la configuración de la sesión de usuario. Permite bloquear el sistema, configurar el tiempo de demora para que el sistema se bloquee automáticamente, cambiar la contraseña del usuario que inició sesión y cerrar la sesión.

Análisis de los sistemas de referencia seleccionados

El análisis de los sistemas para la digitalización de documentos seleccionados ha permitido especificar un conjunto de características que pueden ser estandarizadas y automatizadas durante el proceso de creación de los mismos:

- **Mecanismos de internacionalización de la aplicación**: referida a la posibilidad del sistema de que pueda adaptarse a la configuración de idiomas especificada por el usuario. Esto permite la comercialización del mismo sistema de digitalización en varios idiomas sin necesidad de hacer cambios en la implementación.
- **Configuración del entorno de apariencia de la aplicación:** referida a la posibilidad del usuario de configurar el marco de apariencia con el que desea que sea creada la aplicación. En este caso puede ser configurable también el tema o *look and feel* referido a la fuente, el estilo de los componentes de interfaz de usuario, etc.
- **Configuración de módulos y funcionalidades:** referida a la posibilidad del usuario de establecer los módulos y funcionalidades que tendrá el sistema de digitalización en función del tipo de fondo documental para el cual será creado. Esta característica es muy importante y está determinada por el estudio de fondos documentales con valor legal realizado, cuya cantidad es finita y conocida, y concluye que existen un conjunto de características similares que pueden estar preestablecidas y automatizadas por defecto, aunque cada uno tiene sus propias exigencias en función de su tipología. Se deriva del análisis, la identificación de los procesos críticos dentro del flujo: Recepción y devolución de documentos, Preparación de documentos, Digitalización de documentos, Asociación de metadatos, Control de Calidad, Validación y Exportación; como procesos complementarios:

Administración y configuración, Emisión de Reportes, Gestión de recursos y Gestión de la Producción; y finalmente como proceso opcional Gestión de Almacenes Temporales.

- **Funcionalidades de administración y configuración:** referida a la existencia de un conjunto de funcionalidades de administración y configuración en estos sistemas que se encuentran dispersas, sin embargo pueden ser unificadas y automatizadas en función de robustecer mucho más una aplicación desde el propio proceso de creación.
- **Gestión de certificados digitales:** la mayoría de las aplicaciones utilizan mecanismos de firma digital para garantizar que el objeto digital ha sido obtenido desde un proceso que le otorga valor legal. Consecuentemente la gestión de certificados digitales es una característica importante que puede ser automatizada e incorporada.
- **Características tecnológicas:** la diversidad tanto tecnológica como arquitectónica de las soluciones analizadas constituye una fuente para el análisis y selección de la arquitectura y tecnologías de desarrollo de los sistemas de digitalización que pueden ser establecidas desde el proceso de creación de los mismos.

1.5 Metodología de desarrollo de software

Para guiar el proceso de desarrollo de software es importante la selección de una metodología que oriente y establezca las pautas que deben cumplirse en aras de alcanzar los parámetros de calidad esperados. Existen varias metodologías de software reconocidas internacionalmente que surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un producto de software determinado. De manera general, una metodología es un conjunto de componentes que especifican:

- ¿Cómo se debe dividir un proyecto en etapas?
- ¿Qué tareas se llevan a cabo en cada etapa?
- ¿Qué salidas se producen y cuándo se deben producir?
- ¿Qué restricciones se aplican?
- ¿Qué herramientas se van a utilizar?
- ¿Cómo se gestiona y controla un proyecto?

En el caso de DigiPRO, como se describe en su proyecto técnico (34), se establece el uso de la metodología Rational Unified Process para el desarrollo de las etapas del proyecto, aunque la elaboración de los

artefactos del Expediente de Proyecto está regida bajo las normas establecidas por el Programa de Mejora CMMI Nivel 2 que enfrenta la UCI actualmente.

Rational Unified Process (RUP):

Es una metodología orientada a procesos, define un ciclo de vida iterativo incremental por casos de uso y centrado en la arquitectura. Es un proceso de Ingeniería del Software que proporciona una visión disciplinada para la asignación de tareas y responsabilidades en las organizaciones de desarrollo de software (10).

Ventajas:

- Provee un entorno de proceso de desarrollo configurable, basado en estándares.
- Permite tener claro y accesible el proceso de desarrollo que se sigue.
- Permite ser configurado a las necesidades de la organización y del proyecto.
- Provee a cada participante con la parte del proceso que le compete directamente, filtrando el resto.

Como se muestra en la Figura 4, el ciclo de vida de la metodología RUP está dividido en cuatro fases (10):

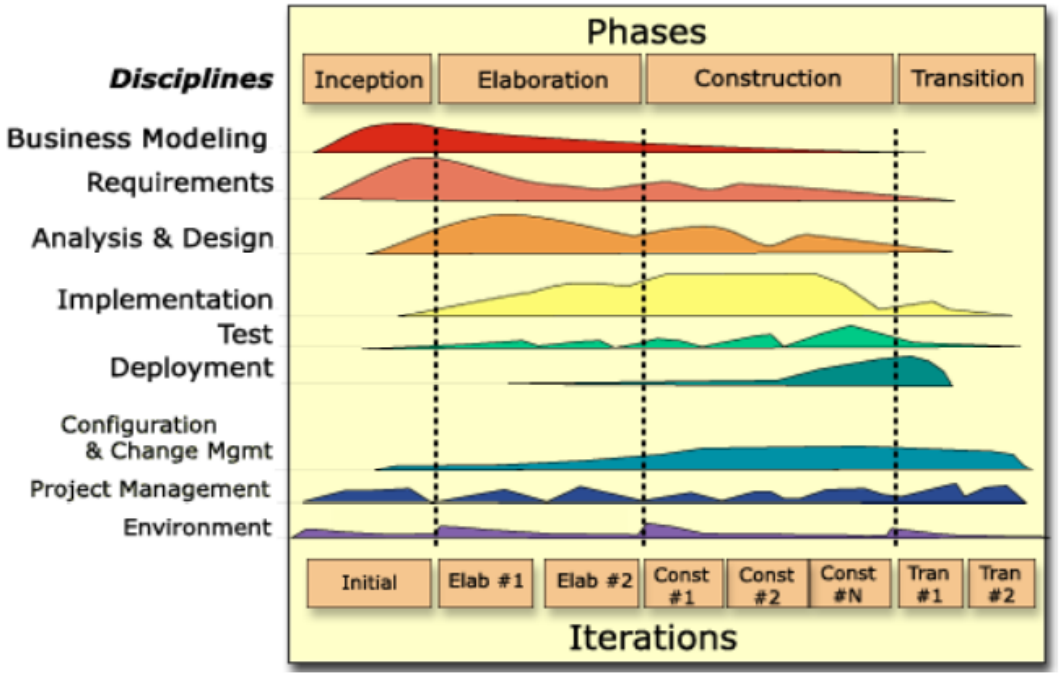

Figura 4 Fases y flujos del RUP.

La presente investigación se enmarca principalmente en las fases de Elaboración y Construcción desarrollando los flujos de trabajo Análisis y Diseño, Implementación y Prueba.

El proceso de desarrollo estará guiado por una adaptación de la Metodología RUP establecida previamente por el equipo a partir de las necesidades propias del entorno del proyecto e influenciada por el cumplimiento de las pautas definidas por el Proceso de Mejora UCI para la elaboración de la documentación.

1.6 Tecnologías para el desarrollo de la propuesta de solución

Durante el proceso de desarrollo la selección de la tecnología es uno de las actividades fundamentales de las que dependen el resto de las etapas. A continuación se fundamenta la selección de las tecnologías utilizadas en el desarrollo de la propuesta:

Lenguaje Unificado de Modelado (UML) 2.0:

El modelado es esencial en la construcción de software para comunicar la estructura de un sistema complejo, especificar el comportamiento deseado del sistema, comprender mejor lo que se construye y descubrir oportunidades de simplificación y reutilización.

El UML es el lenguaje estándar especificado por el Grupo de Gestión de Objetos (OMG, por sus siglas en inglés) para visualizar, especificar, construir y documentar los artefactos de un sistema. Ofrece un estándar para describir los modelos, incluyendo aspectos conceptuales como procesos de negocio, funciones del sistema, expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. En UML es necesario lograr modelos enriquecidos a fin de que las transformaciones automáticas puedan ser soportadas y logradas en su totalidad. La ventaja principal del UML es que unifica distintas notaciones previas como se muestra en la Figura 5.

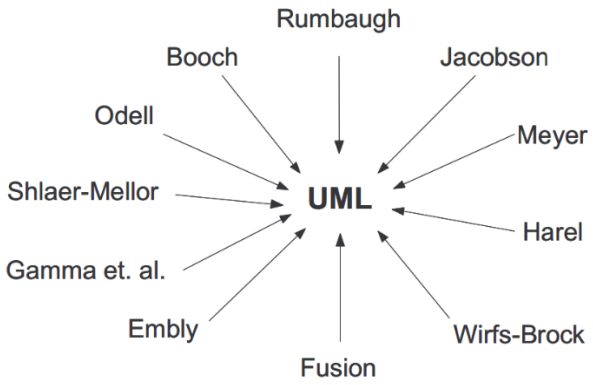

Figura 5 Notaciones precedentes al UML.

El uso del UML también trae consigo determinados inconvenientes como la falta de integración con otras técnicas, es un lenguaje excesivamente complejo (y no está del todo libre de ambigüedades): según (35) , “el 80% de los problemas puede modelarse usando alrededor del 20% de UML”.

Este lenguaje es viable para la modelación de los elementos del diseño debido a que permite la creación de los siguientes diagramas, artefactos necesarios en la solución que se desarrolla:

- Diagrama de Casos de uso: se utilizan en el modelado del sistema desde el punto de vista de sus usuarios para representar las acciones que realiza cada tipo de usuario.
- Diagrama de Clases: muestra un conjunto de clases y sus relaciones.
- Diagrama de Componentes: representa aspectos físicos del sistema.
- Diagrama de Despliegue: muestra la configuración del sistema en tiempo de ejecución.

Herramienta CASE: Visual Paradigm Suite Windows 5.0 for UML:

Visual Paradigm for UML (VP-UML) es un potente modelador visual UML y herramienta CASE (Ingeniería de Software Asistida por Computación), es multiplataforma y fácil de usar. VP-UML provee a los desarrolladores de software una potente plataforma de desarrollo para construir aplicaciones de calidad en poco tiempo. Facilita una excelente interoperabilidad con otras herramientas CASE y la mayoría de los IDEs líderes en el desarrollo, destacándose entre ellos el NetBeans. Soporta el ciclo de vida completo del desarrollo de software: Análisis y Diseño, Implementación y Pruebas.

VP-UML soporta los principales estándares de modelado tales como Unified Modeling Language (UML) 2.4, SoaML, SysML, etc. Es compatible con los equipos de desarrollo de software en la captura de requisitos, la planificación (análisis de casos de uso), ingeniería de código, modelado de clases, modelado de datos, etc. (36).

Algunos elementos que caracterizan esta herramienta son los siguientes:

- Disponibilidad en múltiples plataformas (Windows, Linux).
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Disponibilidad de múltiples versiones, con diferentes especificaciones.
- Soporta el diseño de aplicaciones Web.
- Generación de código para Java y exportación como HTML.

- Soporte de UML versión 2.1.
- Modelado colaborativo con CVS y Subversión (control de versiones).
- Importación y exportación de ficheros XMI.

El uso de esta herramienta, frente a otras de su tipo como Rational Rose (37), está dado por el conocimiento que tiene sobre ella el equipo de desarrollo, las facilidades que brinda para el trabajo colaborativo, su portabilidad y robustez.

Lenguaje de programación JAVA:

Un lenguaje de programación posee una cierta estructura sintáctica y semántica; estos le especifican a la computadora con qué datos debe operar, como debe almacenarlos o transmitirlos y qué acciones debe ejecutar ante distintas circunstancias.

Esta tecnología permite el uso subyacente de punteros, como herramientas, juegos y aplicaciones de negocios. Java se ejecuta en dispositivos móviles y aparatos de televisión a escala global. Es soportado por diversas plataformas ya sean libres o propietarias y brinda soporte a una gran cantidad de aplicaciones que lo usan como plataforma.

Un componente esencial es la máquina virtual de Java (JVM), de gran importancia para el soporte de las aplicaciones construidas con este lenguaje. Permite alcanzar la abstracción de la plataforma de hardware y sistema operativo por lo que todos los programas Java son multiplataforma. Además, garantiza la optimización centralizada de determinadas funciones para su uso por las aplicaciones.

Framework para construcción de interfaces: Swing:

Es un conjunto de clases de java que simplifica la construcción de aplicaciones de escritorio. Permite tener un sistema de ventanas y componentes gráficos, independiente del sistema operativo y las bibliotecas de diseño gráfico que se tengan instaladas. Se utiliza para el diseño y construcción de los formularios de las aplicaciones debido a las ventajas que ofrece para el desarrollo de componentes que permiten la visualización y registro de los datos. Permite el diseño de paneles, botones, campos de texto, áreas de texto, listas desplegables, listas de chequeo, etiquetas que facilitan la visualización del contenido en un formulario, así como la recopilación de datos (38).

Entorno Integrado de Desarrollo NetBeans 6.9:

Es un IDE multilenguaje completo y modular, con soporte para Java SE (Java Standard Edition), Java EE (Java Platform Enterprise Edition) y Java ME (Java Mobile Edition). Soporta distintos lenguajes como por ejemplo: Java, C/C++, PHP, Python y JavaScript. Funciona sobre Unix, Windows, y Mac OS. Soporta todas

las tecnologías definidas por el equipo de desarrollo para la implementación del sistema, especialmente el hecho de que brinda su plataforma de desarrollo para construir aplicaciones modulares.

Plataforma de desarrollo NetBeans 6.9:

Su objetivo es simplificar drásticamente el desarrollo de aplicaciones de escritorio, proporcionando una serie de técnicas, patrones y componentes Swing (39). Ofrece al desarrollador un framework transparente, extensible, libre y de código abierto que garantiza un conjunto de requisitos funcionales muy afines a las aplicaciones de escritorio como:

- Proporcionar interfaces de usuario consistentes.
- Proporcionar mecanismos de extensibilidad de las aplicaciones.
- Permitir la visualización y recopilación de los datos.
- Permitir realizar ajustes de configuración.
- Proveer sistema de ayuda, funciones de actualización en línea.
- Ofrecer facilidades de portabilidad en diferentes sistemas operativos.
- Proporcionar facilidades de usabilidad para el acceso a las funcionalidades del sistema.
- Registrar y visualizar trazas.

Las principales características de esta plataforma son:

- Desarrollo modular de las aplicaciones: dentro de la plataforma se concibe un módulo como un componente de software autónomo con el cual otros módulos pueden comunicarse a través de interfaces bien definidas detrás de las cuales se oculta su implementación. Las responsabilidades granulares bien definidas se encapsulan dentro de un módulo lo que permite mayor especificidad de las responsabilidades y mejores facilidades para el mantenimiento y la realización de cambios. Otra potencialidad que brinda la plataforma es la posibilidad de activar o desactivar dinámicamente los módulos en tiempo de ejecución, aspecto de gran importancia para aplicaciones que administran diferentes niveles de acceso de usuario o que gestionan subsistemas de gran tamaño que deben ser cargados solo cuando se utilizan.
- Sistema de ventanas muy flexible: provee un sistema que permite la flexibilidad para crear ventanas en el área de trabajo que puedan abrirse, cerrarse y moverse en tiempo de ejecución lo que brinda facilidades de configuración de la apariencia de la aplicación por el usuario que puede ser persistible una vez que el usuario cierra la aplicación. Además provee un API que permite manejar el sistema de ventanas desde la programación lo que brinda facilidades al desarrollador a la hora de

implementar los requisitos de interfaz de usuario. Utiliza como superclase el TopComponent, derivado de la clase JComponent y de la cual heredan el resto de los tipos de ventanas que gestiona la plataforma. Esta clase proporciona métodos que permiten controlar la ventana y manipular las notificaciones del sistema de eventos de la ventana principal. Otra de las potencialidades es la clase WindowManager que permite controlar todas las ventanas de la plataforma suficientemente, es decir, rara vez es necesario modificarlo para manipular su comportamiento aunque puede ser posible.

- Persistencia de datos integrada a la plataforma: el usuario puede abstraerse del mecanismo de persistencia de datos de manera que solo decide la información que desea guardar y utiliza los mecanismos para extraerla y modificarla.
- Sistema de ayuda integrado: permite la creación y configuración de la ayuda de manera dinámica y ágil.
- Notificaciones: permite la notificación de mensajes estandarizados y configurables en dependencia de las necesidades del programador.
- Incorpora barras de estado, barras de menús, barras de herramientas y de manera general un área de trabajo muy flexible y común a las aplicaciones de escritorio.

Biblioteca para desarrollo de pruebas: JUnit:

Es una biblioteca de código abierto, liberada bajo la Licencia Pública Común versión 1.0. Es el framework más usado para el desarrollo de pruebas de unidad de software desarrollado con Java. Las pruebas de unidad ayudan a comprobar el correcto comportamiento del código así como a detectar fallos tanto de comportamiento como de concepto (40). Su funcionamiento se basa en la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera. Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente. Por su gran éxito esta herramienta ha sido incorporada a varios IDE, entre los cuales se destaca NetBeans.

1.7 Conclusiones parciales

El análisis de los tipos de herramientas utilizadas durante el proceso de construcción de software permitió caracterizar la solución deseada como una plataforma de desarrollo debido a sus características tecnológicas (dependencia de la Máquina Virtual de Java y soporte sobre la Plataforma NetBeans), integra

componentes para la interacción con dispositivos de hardware, constituye un software para la creación y configuración de productos de digitalización y presenta un conjunto de documentación asociada que guía el proceso.

Los sistemas Digipyrus, DigiDAP y CDA constituyen aportes significativos a las funcionalidades y características arquitectónicas que debe estandarizar la plataforma DigiPRO para la construcción de sistemas de digitalización.

Los sistemas DIGIBÍS, Microfile y XEROX estudiados tienen una aproximación amplia a la solución deseada pero no pueden ser parte de la propuesta porque son sistemas costosos y no garantizan la obtención de objetos digitales con valor legal.

El proceso de desarrollo estará guiado por una adaptación de la Metodología RUP establecida previamente por el equipo a partir de las necesidades propias del entorno del proyecto e influenciada por el cumplimiento de las pautas definidas por el Proceso de Mejora UCI para la elaboración de la documentación.

El análisis sobre las tecnologías de desarrollo a utilizar determinó la selección como herramienta CASE para la obtención de los elementos del diseño el Visual Paradigm 5.0, el UML 2.0 como lenguaje de modelado, el lenguaje de programación será Java, Swing como herramienta para la construcción de interfaces; el Entorno de Desarrollo Integrado NetBeans 6.9.1 por el soporte relacionado a la Plataforma NetBeans que sustentará la propuesta de solución y como biblioteca para el desarrollo de pruebas se usará JUNIT.

CAPÍTULO 2 Características y diseño del sistema

2.1. Introducción

En este capítulo se presentan las características del subsistema Plataforma, los requisitos funcionales y no funcionales que debe satisfacer la solución y la especificación del sistema en función de los casos de uso del análisis. Se muestra además, la descripción de la arquitectura que soporta la plataforma, los principales resultados del diseño expresados en los diagramas y la descripción de las clases, así como la validación del trabajo realizado en esta etapa a partir de métricas de diseño de clases.

2.2. Características del sistema

El subsistema Plataforma tiene como objetivo general: agilizar y estandarizar el proceso de creación de sistemas para la automatización de Centros de Digitalización flexibles a las exigencias del negocio de los diferentes entornos empresariales, permitiendo la creación y configuración de sistemas para la generación de objetos digitales con valor legal.

Está estructurado en dos módulos, como se muestra en la Figura 6.

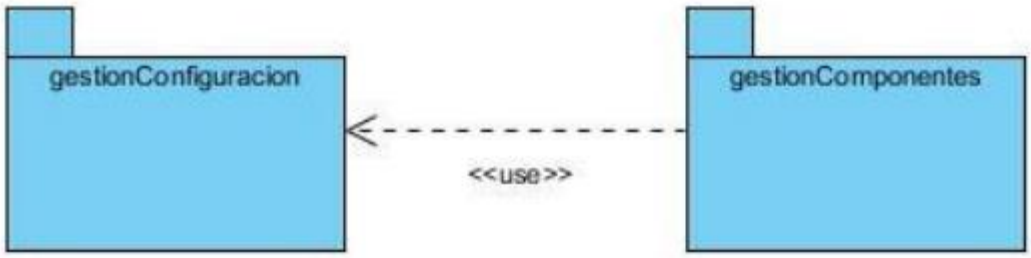

Figura 6 Módulos del subsistema Plataforma.

Módulo Gestión de Configuración: permite gestionar la configuración de todos los componentes, necesaria para la creación de proyectos de digitalización.

Procesos que componen el módulo:

- Gestión de apariencia: permite a los usuarios adicionar, modificar, habilitar o deshabilitar los marcos y temas que componen la apariencia de los productos de software que pueden ser creados a través de la plataforma.
- Gestión de idiomas de la plataforma: permite a los usuarios adicionar, modificar, habilitar o deshabilitar los idiomas que pueden ser aplicados a la plataforma.

- Gestión de idiomas de productos: permite a los usuarios adicionar, modificar, habilitar o deshabilitar los idiomas que pueden ser aplicados a los productos de software que pueden ser creados a través de la plataforma.
- Selección de idiomas: permite a los usuarios seleccionar un idioma habilitado previamente en el que serán mostrados los mensajes y etiquetas de la plataforma.
- Gestión de certificados digitales: permite a los usuarios adicionar, deshabilitar, habilitar o generar certificados digitales para la firma digital de los componentes de la plataforma.
- Gestión de módulos: permite a los usuarios gestionar los módulos y funcionalidades de la plataforma para la creación de productos.
- Notificación de mensajes: muestra a los usuarios eventos significativos del sistema que se requiera notificar, ya sean de información, advertencia o error.
- Gestión de accesos directos: permite a los usuarios configurar la barra de accesos directos a partir de las funcionalidades que proporciona la plataforma.

Módulo Gestión de componentes: permite agrupar y configurar los componentes durante la creación de sistemas para la digitalización de documentos.

Procesos que componen el módulo:

- Creación de proyectos: permite a los usuarios la creación de un nuevo proyecto con sus parámetros iniciales.
- Configuración de i18N (Internacionalización de idioma): permite a los usuarios adicionar o eliminar idiomas a un producto de software durante el proceso de creación.
- Configuración de módulos: permite a los usuarios adicionar o eliminar módulos de un proyecto existente de acuerdo a las restricciones del negocio.
- Configuración de apariencia: permite a los usuarios seleccionar un marco de apariencia y un tema para un producto de software durante su creación.
- Configuración de políticas adicionales: permite a los usuarios realizar configuraciones adicionales a un producto de software durante su creación.
- Generación del producto: permite a los usuarios generar el producto final a partir de todas las configuraciones realizadas.

2.3. Especificación de los requisitos de software

Es el proceso de averiguar, normalmente en circunstancias difíciles, lo que se debe construir. El propósito fundamental del flujo de trabajo de los requisitos es guiar el desarrollo hacia el sistema correcto. Esto se consigue mediante una descripción de los requisitos del sistema (es decir, las condiciones o capacidades que el sistema debe cumplir) suficientemente buena como para que pueda llegarse a un acuerdo entre el cliente (incluyendo a los usuarios) y los desarrolladores sobre qué debe y qué no debe hacer el sistema (10).

2.3.1 Requisitos funcionales

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer. Estos dependen del tipo de software que está siendo desarrollado, los usuarios previstos del software y el enfoque general adoptado por la organización al escribir requisitos (41).

La propuesta de solución contempla 21 requisitos funcionales que se encuentran descritos en el documento Especificación de requisitos de software del Subsistema Plataforma (42). A continuación se muestran los más importantes agrupados en el módulo Gestión de componentes:

RF_43 Crear nuevo proyecto

El sistema debe permitir al administrador crear un nuevo proyecto.

RF_44 Seleccionar marco de apariencia

El sistema debe permitir al administrador seleccionar un marco de apariencia para aplicarlo al proyecto en creación.

RF_45 Deseleccionar marco de apariencia

El sistema debe permitir al administrador deseleccionar un marco de apariencia.

RF_46 Visualizar los marcos de apariencia

El sistema debe ser capaz, de forma autónoma, de visualizar el listado de marcos de apariencia que pueden ser aplicados a un proyecto en creación.

RF_47 Aplicar tema de apariencia

El sistema debe permitir al administrador aplicar un tema al proyecto en creación.

RF_48 Visualizar los módulos, funcionalidades y variantes existentes

El sistema debe ser capaz, de forma autónoma, de visualizar el listado de módulos y funcionalidades con sus variantes que pueden ser adicionados a un proyecto en creación.

RF_49 Adicionar variante a proyecto en creación.

El sistema debe permitir al administrador adicionar variantes de implementación de una funcionalidad al proyecto que se está creando.

RF_ 50 Eliminar variante de funcionalidad

El sistema debe permitir al administrador eliminar una variante de implementación de una funcionalidad del proyecto que se está creando.

RF_51 Visualizar los idiomas existentes en la plataforma

El sistema debe ser capaz, de forma autónoma, de visualizar el listado de idiomas existentes en la plataforma durante la creación de un proyecto.

RF_52 Adicionar idioma

El sistema debe permitir al administrador adicionar idiomas a un proyecto en creación.

RF_53 Eliminar idiomas

El sistema debe permitir al administrador eliminar uno o varios idiomas previamente adicionados a un proyecto durante su creación.

RF_54 Visualizar las políticas de configuración establecidas por defecto

El sistema debe ser capaz, de forma autónoma, de visualizar el listado de políticas de configuración establecidas por defecto.

RF_55 Modificar valor de políticas de configuración

El sistema debe permitir al administrador modificar el valor de las políticas de configuración.

RF_56 Visualizar la configuración establecida

El sistema debe ser capaz, de forma autónoma, de visualizar todos los detalles de la configuración establecida por el administrador durante la creación de un proyecto.

RF_57 Comprobar la integridad y autenticidad de los datos

El sistema debe ser capaz, de forma autónoma, de comprobar la autenticidad e integridad de los datos registrados y emitir un mensaje de información sobre los resultados de la validación.

RF_58 Imprimir documentación

El sistema debe permitir al administrador imprimir la documentación referente al proyecto.

RF_59 Generar producto

El sistema debe permitir al administrador generar el instalador del producto a partir de las configuraciones establecidas previamente.

RF_60 Abrir proyecto

El sistema debe permitir al administrador abrir un nuevo proyecto guardado anteriormente.

RF_61 Guardar proyecto

El sistema debe permitir al administrador guardar un nuevo proyecto en el que está trabajando actualmente.

RF_62 Duplicar proyecto

El sistema debe permitir al administrador duplicar un proyecto guardado anteriormente.

RF_63 Establecer espacio de trabajo

El sistema debe permitir al administrador establecer el espacio de trabajo en el que se guardarán todos los datos procesados por la plataforma.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable (43). La propuesta de solución contempla 31 requisitos no funcionales que se encuentran descritos en el documento Especificación de requisitos no funcionales del Subsistema Plataforma (44). A continuación se muestran los más importantes agrupados por categorías:

Usabilidad

RnF 02. Requisito de Usabilidad 2. Desarrollar una aplicación informática de escritorio que facilite la construcción de sistemas con esta característica.

La plataforma será una aplicación informática de escritorio para facilitar la construcción de sistemas informáticos de escritorio.

RnF 03. **Requisito de Usabilidad 3. Permitir la construcción de sistemas para la obtención de objetos digitales con valor legal a través de la digitalización de documentos.**

La aplicación informática tendrá el objetivo de facilitar el desarrollo y construcción de sistemas para la obtención de objetos digitales con valor legal a través de la digitalización de documentos.

RnF 04. Requisito de Usabilidad 4. Cumplir con las pautas de diseño de las interfaces.

El sistema debe tener una interfaz gráfica uniforme incluyendo: pantallas, menús y opciones. Las pautas de diseño serán definidas por el equipo de diseño gráfico y se realizarán siguiendo los lineamientos de la arquitectura de información.

RnF 05. Requisito de Usabilidad 5. Agrupar vínculos y botones por grupos funcionales.

La consistencia de la interacción entre usuario y sistema estará determinada por el diseño de la interfaz de usuario que mantendrá los elementos como menús, banners y zona de trabajo, en posiciones fijas, además de la mayor uniformidad posible entre cuadros de texto y botones.

El sistema debe ser de uso intuitivo, de tal forma que se reduzcan los tiempos de entrenamiento, soporte y prueba por parte del usuario. La agrupación de los botones por funcionalidad determinará además la capacidad de componer la interfaz de acuerdo a las funciones requeridas por un rol determinado.

RnF 07. **Requisito de Usabilidad 7. Utilizar campos de selección en la interfaz en los casos que sea posible.**

El sistema debe facilitar la entrada de datos a los usuarios, presentando campos de selección que permitan escoger los valores predeterminados por el sistema. Estos campos contendrán los valores posibles con los que se podrá llenar un determinado elemento en la interfaz, haciendo que el proceso de llenado de datos sea lo más intuitivo posible.

Los elementos mostrados en estos campos de valores deben ser configurables, de manera que se puedan cambiar o agregar nuevos.

RnF 09. **Requisito de Usabilidad 9. Instalar en las estaciones de trabajo el software necesario para el correcto funcionamiento del sistema.**

En las estaciones de trabajo clientes deben estar instaladas las siguientes herramientas:

- Máquina Virtual de Java.
- Drivers para interacción con dispositivo de firma digital.

RnF 010. Requisito de Usabilidad 10. Proporcionar características mínimas de hardware a las estaciones de trabajo que permitan el correcto funcionamiento del sistema.

Las características técnicas mínimas de hardware que deben tener las estaciones de trabajo son las siguientes:

- Procesador de doble núcleo a 2.0 GHz.
- 2 GB de RAM, DDR2 667 MHz.
- 160 GB de disco duro, SATA 5400 RPM.
- Adaptador de red Ethernet 1 Gbps.
- Sistema de Energía Ininterrumpida (UPS) 500 Va.

Confiabilidad

RnF 012. Requisito de Confiabilidad 2. Reportar mensajes del sistema.

El sistema deberá reportar todos los mensajes de información, advertencias o errores originados durante las operaciones realizadas por el usuario.

Eficiencia

RnF 013. Requisito de Eficiencia 1. Responder en tiempos aceptables las peticiones que se realicen en el sistema.

El sistema debe ser capaz de dar respuestas a las peticiones con un nivel aceptable de desempeño sin sobrepasar los 4 segundos para cada transacción. Teniendo en cuenta el nivel de concurrencia que pueda existir, debe ser capaz de prestar servicios sin que se afecten los tiempos de respuestas predefinidos.

RnF 014. Requisito de Eficiencia 2. Garantizar el consumo de RAM del sistema en las estaciones de trabajo por debajo de los 80 Mb.

El proceso del sistema no debe consumir más de 80 Mb de RAM en las estaciones de trabajo.

Restricciones de diseño

RnF 017. **Requisito de Restricciones de diseño 1. Desarrollar a través de un modelo orientado a componentes.**

El sistema se construirá basado en el modelo de desarrollo orientado a componentes facilitando así la mantenibilidad y la reutilización de los componentes del sistema.

RnF 018. Requisito de Restricciones de diseño 2. Desarrollar utilizando el lenguaje de programación JAVA, el entorno integrado de desarrollo NetBeans 6.9 y como herramienta CASE Visual Paradigm 8.0.

Para el desarrollo del sistema se utilizarán como lenguaje de programación JAVA debido a las potencialidades que ofrece para construir sistemas de este tipo, como entorno integrado de desarrollo el NetBeans 6.9 debido al soporte amplio que presenta para el uso de este lenguaje y como herramienta CASE el Visual Paradigm debido a la experiencia que posee el equipo de trabajo en su uso.

Requisitos para la documentación de usuarios en línea y ayuda del sistema

RnF 020. **Requisito de Documentación 1. Integrar ayuda al sistema.**

El sistema debe contar con una ayuda integrada que permita al usuario orientarse en cada una de sus interfaces. La ayuda debe brindar una explicación detallada del contenido de la interfaz.

Estándares aplicables

RnF 025. **Requisito de estándares aplicables 2. Elaborar el diseño de interfaces gráficas de usuario y prototipos del sistema de acuerdo a los estándares de diseño definidos por el equipo de desarrollo.**

Para el diseño de interfaces gráficas de usuario y prototipos del sistema se seguirán las pautas establecidas en el documento Estándares para el diseño de Interfaces de Usuario. Consultar documento (45).

RnF 027. **Requisito de estándares aplicables 4. Elaborar los artefactos del Diseño de acuerdo a los estándares de diseño definidos por el equipo de desarrollo.**

Para la elaboración de los artefactos del Diseño se seguirán las pautas establecidas en el documento Estándares de Diseño. Consultar documento (46).

RnF 028. **Requisito de estándares aplicables 5. Elaborar los artefactos de Requisitos de acuerdo a los estándares de especificación de requisitos definidos por el equipo de desarrollo.**

Para la elaboración de los artefactos de Requisitos se seguirán las pautas establecidas en el documento Estándares de Requisitos. Consultar documento (47).

2.4. Especificación de los casos de uso

Un caso de uso (CU) es una secuencia de acciones que el sistema lleva a cabo para ofrecer algún resultado de valor para un actor. El modelo de casos de uso está compuesto por todos los actores y todos los casos de uso de un sistema (35).

2.4.1 Actores del sistema

Un actor es alguien o algo ajeno al sistema que interactúa con el mismo. Los actores representan terceros fuera del sistema que colaboran con él sistema (35). Los actores que interactúan con los casos de uso del sistema se describen a continuación:

Actor	Objetivo
Administrador	Actor encargado de acceder al sistema y realizar las operaciones.
Sistema	Actor que referencia las acciones del sistema sin la intervención del usuario.

2.4.2 Diagramas de casos de uso

“Un diagrama de casos de uso muestra un grupo de casos de uso, los actores y sus relaciones, representan la vista estática de un sistema. Estos, son especialmente importantes en la organización y modelado del comportamiento del sistema” (48) .

La propuesta de solución contempla 16 casos de uso que se encuentran descritos en el documento Especificación de casos de uso del Subsistema Plataforma (49). A continuación se muestran, en la Figura 7 los más importantes referentes al módulo Gestión de componentes:

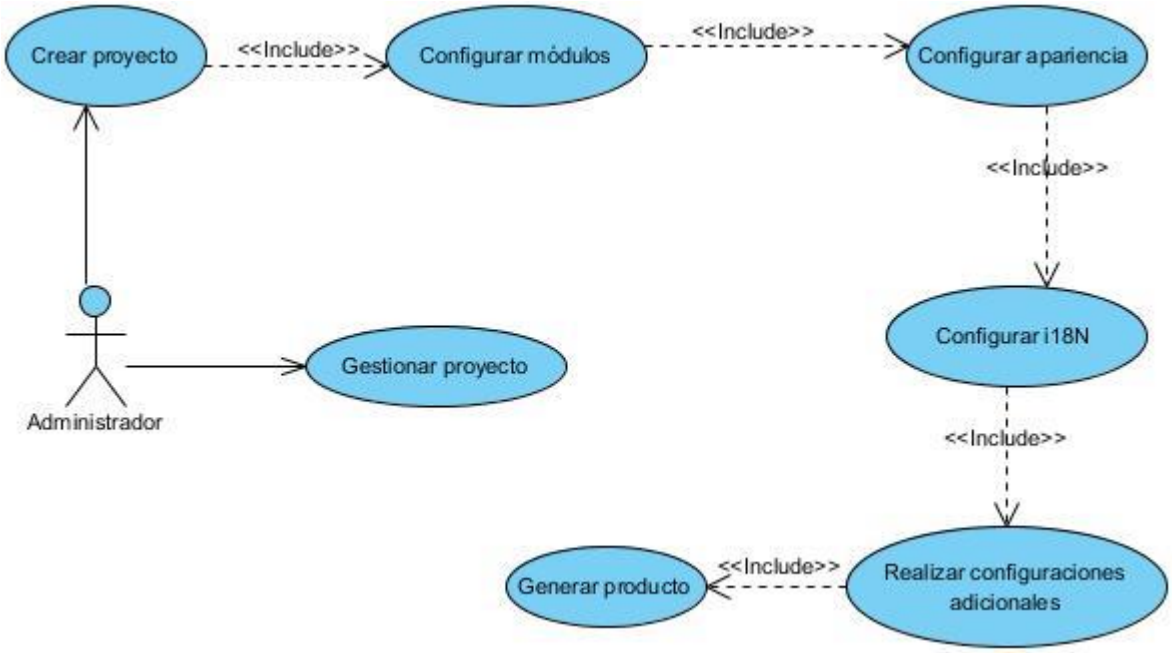

Figura 7 Diagrama de CU Gestión de componentes.

A continuación se describe el CU Crear proyecto, el resto de los CU se encuentran descritos en (49):

Objetivo	Crear un nuevo proyecto cuyo resultado es un producto de software para Centro de Digitalización.
Actores	Administrador: (Inicia) Crea un nuevo proyecto.
Resumen	Permite la creación de un nuevo proyecto.
Complejidad	Baja
Prioridad	Crítico
Precondiciones	La plataforma ha iniciado correctamente.
Postcondiciones	El proyecto se creó.
Flujo de eventos	
Flujo básico Crear Proyecto	

	Actor	Sistema
1	Selecciona la opción “Crear proyecto”.	
2		Muestra el asistente de creación de proyectos en su pantalla inicial para registrar los datos primarios del nuevo proyecto.
3	Registra los datos primarios: - Nombre - Creador - Descripción - Fecha - Propietario - Si desea crearlo desde un proyecto existente - Localización Selecciona la opción “Siguiente”.	
4		Guarda los datos temporalmente y muestra la pantalla “Configurar módulos”. Ver CUS “Configurar módulos”.
5	Termina el caso de uso.	

Flujos alternos		
3.a. Selecciona la opción “Cancelar”		
Actor		**Sistema**
1	Selecciona la opción “Cancelar”.	
2		Descarta los cambios realizados y cierra la interfaz “Crear proyecto”.
3	Termina el caso de uso.	

Relaciones	**CU incluidos**	Configurar módulos.

	CU extendidos	Este caso de uso no tiene casos de uso extendidos.
Requisitos no funcionales	RnF 04, RnF 05.	
Asuntos pendientes	No se identifican posibles mejoras al caso de uso actualmente.	

El prototipo elemental de interfaz gráfica de usuario del CU es el siguiente:

Figura 8 Prototipo funcional para la creación de proyectos.

2.4.3 Patrones de casos de uso

Son comportamientos que deben existir en el sistema, ayudan a describir que es lo que el sistema debe hacer, es decir, describen el uso del sistema y como este interactúa con los usuarios. Estos patrones son utilizados generalmente como plantillas que describen como deberían ser estructurados y organizados los casos de uso. Son patrones que capturan mejores prácticas para modelar casos de uso (50).

Patrón CRUD

CRUD Completo: consiste en un caso de uso para administrar la información (CRUD Información), nos permite modelar las diferentes operaciones para administrar una entidad de información, tales como crear, leer, cambiar y eliminar o dar de baja. Este patrón deberá ser usado cuando todas las operaciones

contribuyen al mismo valor de negocio y todas son cortas y simples. El uso de este patrón se evidencia en los Casos de Uso Gestionar marcos de apariencia, Gestionar certificados digitales y Gestionar módulos.

Patrón Inclusión: se incluye una relación del caso de uso base al caso de uso de inclusión. El último puede ser instalado en sí mismo. El caso de uso base puede ser concreto o abstracto. El uso de este patrón se evidencia en los Casos de Uso Crear proyecto, Configurar módulos y Configurar apariencia.

2.5. Descripción de la arquitectura

La arquitectura de software alude a la estructura general del software y las formas en que la estructura proporciona una integridad conceptual para un sistema. En su forma más simple, la arquitectura es la estructura u organización de los componentes del programa (módulos), la manera en que estos componentes interactúan, y la estructura de datos que utilizan los componentes. En sentido más amplio, sin embargo, los componentes pueden generalizarse para representar elementos importantes del sistema y sus interacciones (43).

La arquitectura de software de la solución propuesta está basada sobre la arquitectura de la Plataforma NetBeans. Este framework permite la construcción del software de forma modular, ofrece implementados los mecanismos de descubrimiento de nuevos módulos (y de actualizaciones de los existentes) desde repositorios remotos, resolución de dependencias, activación/desactivación de módulos en tiempo de ejecución así como la comunicación entre los mismos. También incorpora módulos que garantizan la solución de requisitos funcionales que tienen la mayoría de las aplicaciones de escritorio como Integración a Ayuda, Internacionalización, gestión de funcionalidades y menú de aplicaciones, etc.

Aunque los módulos se distribuyen al mismo nivel en la suite, cada uno pertenece a una capa de implementación de la arquitectura, por lo que puede identificarse la distribución de la misma en 4 capas. Los módulos se distribuyen por capas en la suite, como lo muestra la Figura 9:

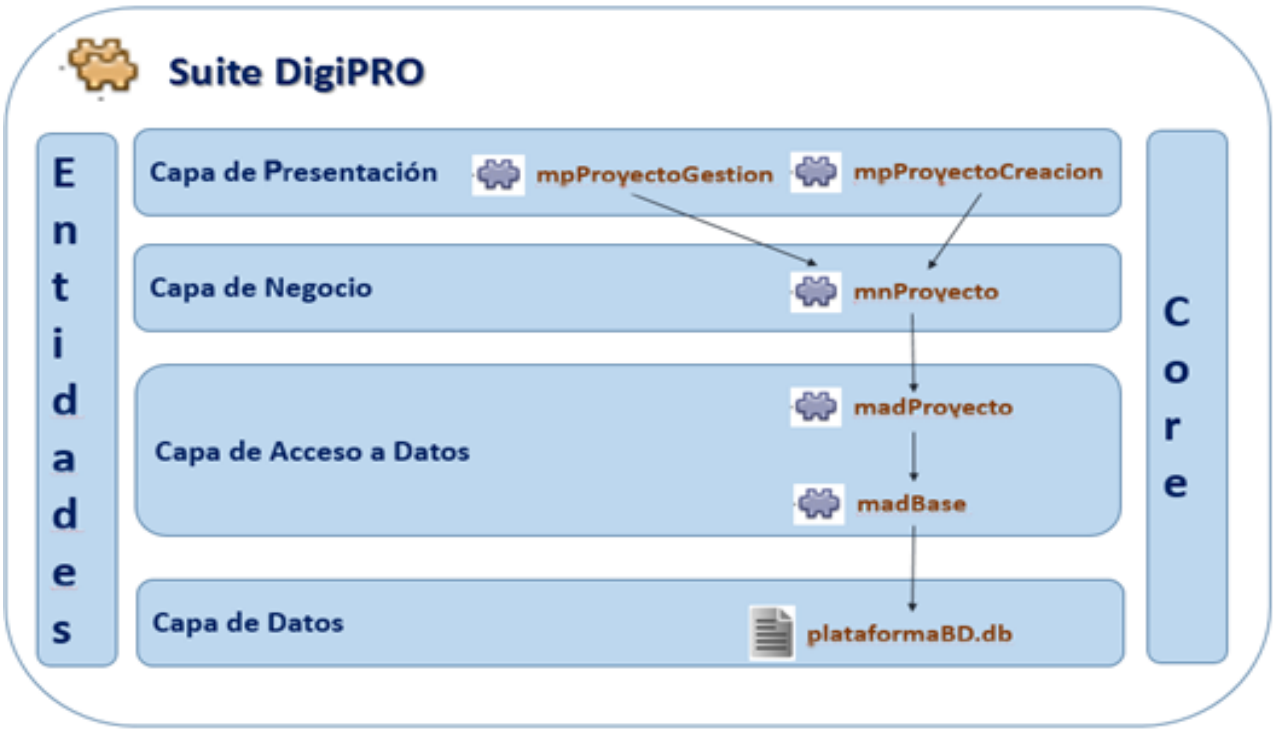

Figura 9 Distribución de componentes en la arquitectura del sistema por capas.

Los elementos se describen a continuación:

Capa de presentación: contiene los módulos que gestionan la presentación y la recopilación de datos desde los formularios del sistema, ya sea en forma de TopComponents o Actions del sistema. Estos interactúan con los módulos de negocio para su funcionamiento.

Capa de negocio: contiene los módulos de negocio del sistema que se encargan de la implementación de la lógica de negocio necesaria para satisfacer los requisitos funcionales. Interactúa con los módulos de la capa de acceso a datos para aquellas funcionalidades que necesitan persistencia o consulta de datos.

Capa de acceso a datos: contiene los módulos encargados de las funcionalidades de acceso a datos que permiten la persistencia y la consulta de los mismos. Contiene el componente madBase que se encarga de gestionar la conexión y operaciones básicas con la base de datos.

Capa de datos: representada por el fichero plataformaBD.db donde se almacenan los datos con los que trabaja el sistema.

Entidades: contiene las entidades de dominio a las que acceden los módulos correspondientes para su funcionamiento.

Core: contiene los módulos del núcleo de la plataforma que garantizan las siguientes funcionalidades:

- mcComponentesIU: componentes de interfaz de usuario que facilitan la visualización y recopilación de datos.
- mcDialogos: permite la visualización de cuadros de diálogos personalizados en la aplicación.
- mcMensajes: permite la visualización de mensajes de Información, Advertencia, Confirmación o Error.
- mcI18N: permite la internacionalización de los módulos del sistema.
- mcExcepciones: facilita el tratamiento de excepciones desde todos los módulos del sistema.
- mcValidacion: permite la validación de los datos de entrada desde la capa de presentación a través de expresiones regulares.

2.6. Diseño

El modelo de diseño se presenta a partir de la realización de los Casos de Uso del diseño, entendido como los diagramas de clases y la descripción de las clases del diseño.

2.6.1 Diagrama de clases

Un diagrama de clases es un tipo de diagrama estático, orientado a objetos, que describe la estructura de un sistema mostrando sus clases. El diagrama de clases incluye mucha más información como la relación entre un objeto y otro, la herencia de propiedades de otro objeto y conjuntos de operaciones/propiedades que son implementadas para una interfaz gráfica (51). Presenta las clases del sistema con sus relaciones estructurales y de herencia.

El subsistema Plataforma cuenta con 92 clases del diseño. La realización de los diagramas del diseño se encuentra en el modelo de diseño del expediente de proyecto (52). A continuación se presentan las relacionadas al CU Crear proyecto:

2.6.2 Descripción de las clases

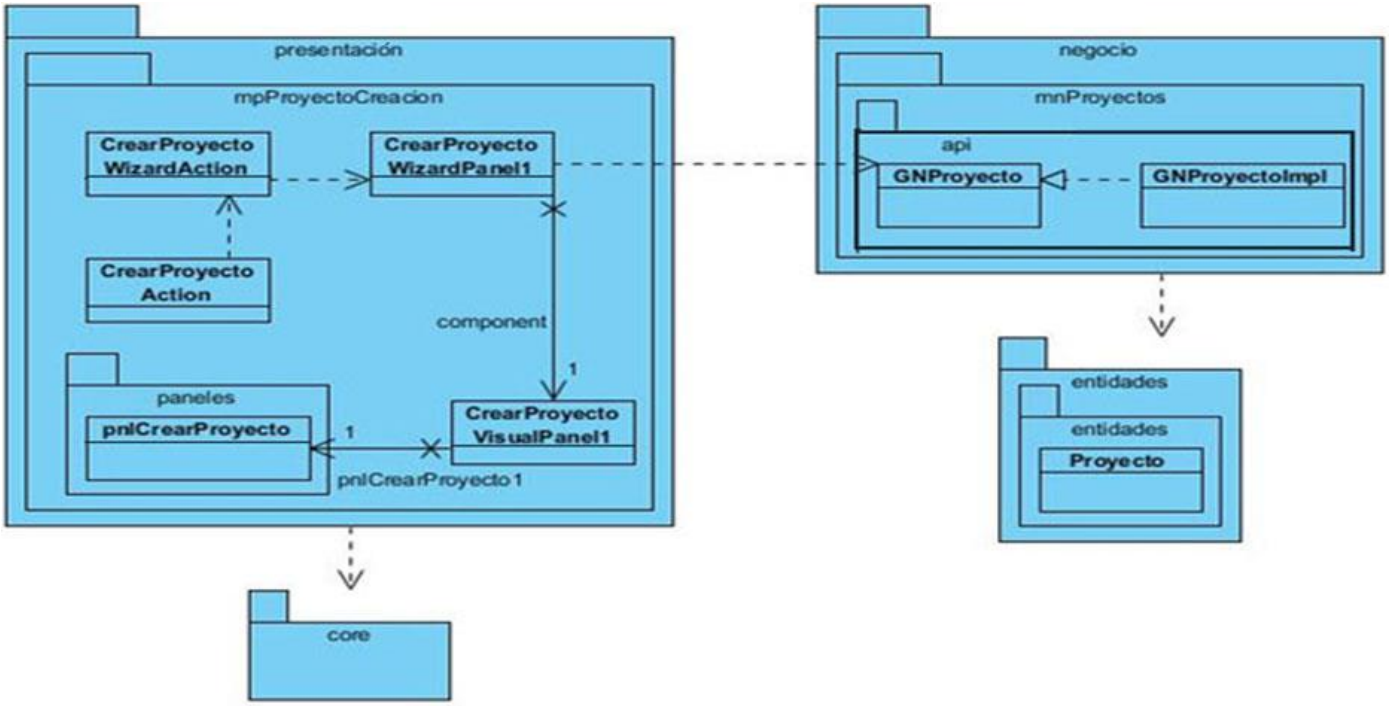

Figura 10 Diagrama de Clases Crear Proyecto.

CrearProyectoVisualPanel1: clase que muestra el formulario pnlCrearProyecto.

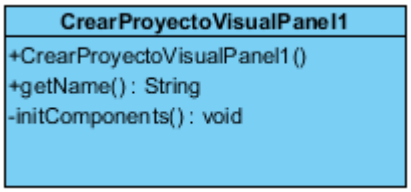

Figura 11Clase CrearProyectoVisualPanel1.

CrearProyectoWizardPanel1: clase que gestiona las funcionalidades del formulario CrearProyectoPanel1.

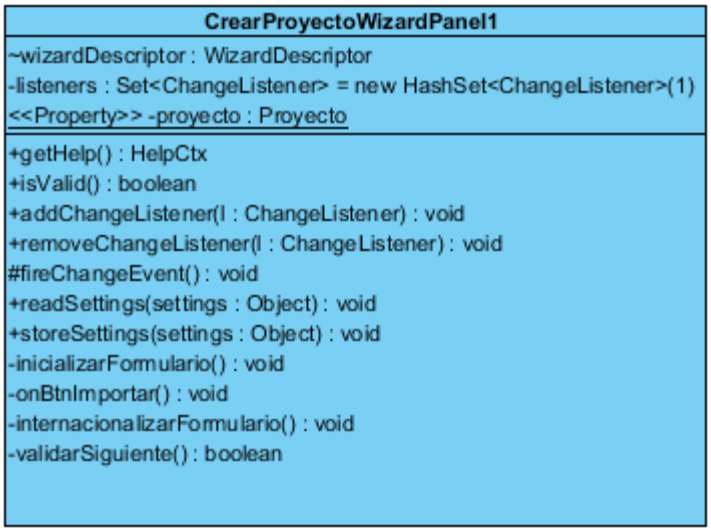

Figura 12 Clase CrearProyectoWizarPanel1.

CrearProyectoAction: Clase que describe la acción Crear proyecto.

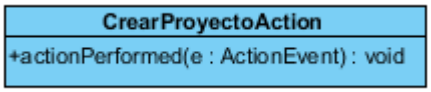

Figura 13 ClaseCrearProyectoAction.

CrearProyectoWizardAction: Clase que gestiona el wizard para Crear Proyecto.

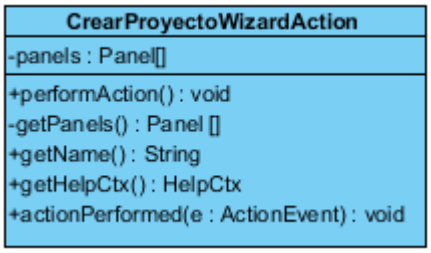

Figura 14 CrearProyectoWizardAction.

pnlCrearProyecto: Clase que representa el formulario de la funcionalidad Crear proyecto.

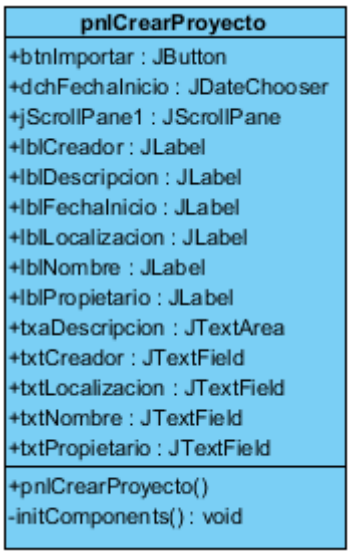

Figura 15 Clase pnlCrearProyecto.

GNProyecto: Interfaz que gestiona la lógica de negocio para la entidad de dominio Proyecto.

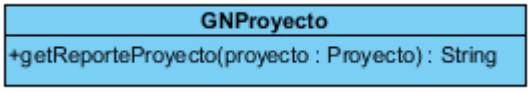

Figura 16 Clase GNProyecto.

GNProyectoImpl: Clase que gestiona la lógica de negocio para la entidad de dominio Proyecto. Implementa la interfaz GNProyecto.

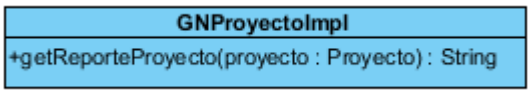

Figura 17 Clase GNProyectoImpl.

Proyecto: Entidad de dominio que representa un proyecto.

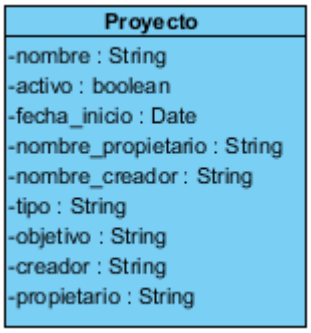

Figura 18 Clase Proyecto.

2.6.3 Patrones de diseño

Patrones Grasp

Patrón Creador: el patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El objetivo fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento.

Patrón Experto: se basa en asignar una responsabilidad a la clase que cuenta con la información necesaria para cumplir dicha responsabilidad. Este patrón permite conservar el encapsulamiento, ya que los objetos se valen de su propia información para realizar lo que se le oriente.

Patrón Bajo Acoplamiento: este patrón asigna una responsabilidad para mantener el bajo acoplamiento. La idea es tratar de que una clase no dependa de muchas otras, así esa clase no tendrá muchas dependencias, lo que facilitará la reutilización de la misma y se reducirá el impacto de los cambios.

Patrón Alta Cohesión: una clase con alta cohesión tiene un número relativamente pequeño de métodos, con funcionalidades altamente relacionadas, y no realiza mucho trabajo, colaborando con otros objetos para compartir el esfuerzo si la tarea es extensa. Las clases con alta cohesión son relativamente fáciles de mantener, entender y reutilizar.

Patrón Controlador: para manejar y controlar los eventos del sistema. Con la utilización de este patrón se logra separar la lógica de negocio de la capa de presentación. De esta manera se logra un mayor control sobre el sistema y se favorece la reutilización de código. (33)

Patrones Gang of Four (GoF)

Los patrones de diseño GoF se clasifican en 3 categorías basadas en su propósito:

- Creacionales: tratan con las formas de crear instancias de objetos. El objetivo de estos patrones es de abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados.
- Estructurales: los patrones estructurales describen cómo las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades. Estos objetos adicionados pueden ser incluso objetos simples u objetos compuestos.
- Comportamiento: los patrones de comportamiento ayudan a definir la comunicación e iteración entre los objetos de un sistema. El propósito de este patrón es reducir el acoplamiento entre los objetos.

Decorador / Decorator (categoría: Estructurales): es el patrón que se emplea para mejorar la funcionalidad de una clase a la que envuelve, que ya funciona sin él, pero no altera las salidas que esta produce. Se puede permitir añadir a los componentes visuales propiedades (por ejemplo, un borde) o comportamientos (por ejemplo, una barra de desplazamiento).

Solitario / Singleton (categoría: Creacionales): patrón que restringe la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto.

Fachada / Facade (categoría: Estructurales): proporcionar una interfaz unificada de alto nivel que, representando a todo un subsistema, facilite su uso. La “fachada” satisface a la mayoría de los clientes, sin ocultar las funciones de menor nivel a aquellos que necesiten acceder a ellas.

2.7 Conclusiones parciales

En este capítulo se abordaron los elementos fundamentales correspondientes a las fases de planificación y diseño del sistema informático que se propone, apoyándose en las técnicas relacionadas a la metodología RUP para estas fases de desarrollo. Fueron identificadas 21 funcionalidades que el sistema debe cumplir mientras que los requisitos no funcionales cubren clasificaciones como usabilidad, confiabilidad, soporte y seguridad. El presente capítulo permitió definir la estructura con que contará el subsistema plataforma, dando como resultado la creación de dos módulos principales denominados gestión de Configuración y gestión de Componentes definiendo cada una de las características que tendrán ambos módulos. Se determina también que la propuesta de solución contempla 16 casos de uso los cuales satisfacen las necesidades de los usuarios del sistema. Se definieron patrones de casos de uso como el patrón CRUD

Completo y el Patrón Inclusión, los cuales definen los comportamientos que deben existir en el sistema. El capítulo permitió llegar a la conclusión de que el uso de la Plataforma NetBeans permite implementar todas las funcionalidades y requerimientos del subsistema y también desarrollar el mismo dividido en cuatro capas fundamentales de acuerdo a su uso. También se realizaron los diagramas de clases de la aplicación y el análisis de cada uno de los componentes del mismo, lo que permitió la distribución de las mismas en cada una de las capas de la arquitectura.

CAPÍTULO 3 Implementación y validación de la solución

3.1 Introducción

En este capítulo se describe el proceso de implementación de la propuesta presentando los elementos del modelo de despliegue y diagramas de componentes. Además se muestran los resultados del análisis del proceso de verificación y validación de la solución con la aplicación de pruebas de software y la validación de la idea a defender presentada en la investigación.

3.2 Modelo de despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una gran influencia en su diseño (10). En la Figura 19 se muestra una vista del modelo de despliegue de la plataforma:

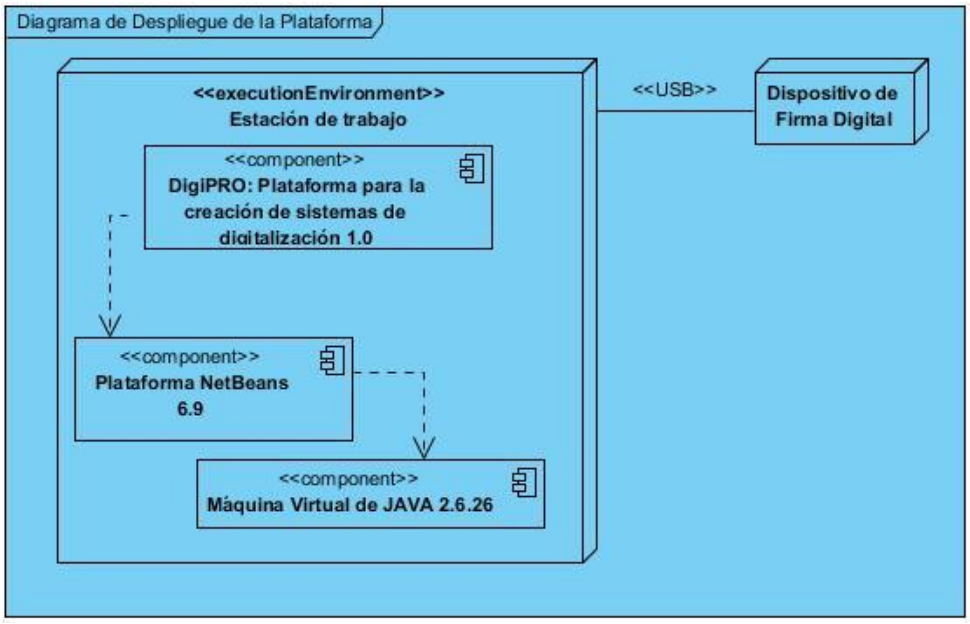

Figura 19 Diagrama de despliegue.

Los componentes del diagrama de despliegue son:

- **Estación de trabajo:** nodo con capacidad de procesamiento donde se distribuyen los componentes de la aplicación.
- **Dispositivo de firma digital:** dispositivo para la firma digital de los componentes de la aplicación.

- **Componentes de software desplegados:** Plataforma DigiPRO, Plataforma NetBeans y Máquina Virtual de Java.

3.3 Implementación

Los principales artefactos obtenidos durante la implementación fueron el Modelo de Despliegue de la solución y el modelo de implementación que contiene los diagramas de componentes. Se presentan además los estándares de codificación utilizados durante la implementación de la propuesta.

3.3.1 Diagramas de componentes

Un diagrama de componentes muestra las dependencias lógicas entre componentes de software, sean éstos componentes: fuentes, binarios o ejecutables, ilustran las piezas del software, controladores embebidos, etc. Prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema, es decir para describir la vista de implementación estática de un sistema (53).

El subsistema Plataforma cuenta con 32 componentes o módulos distribuidos en las capas de la arquitectura. A continuación se presenta el diagrama de componentes del caso de uso Crear proyecto, el resto de los diagramas de componentes se muestra en el ¡Error! No se encuentra el origen de la referencia.**.**

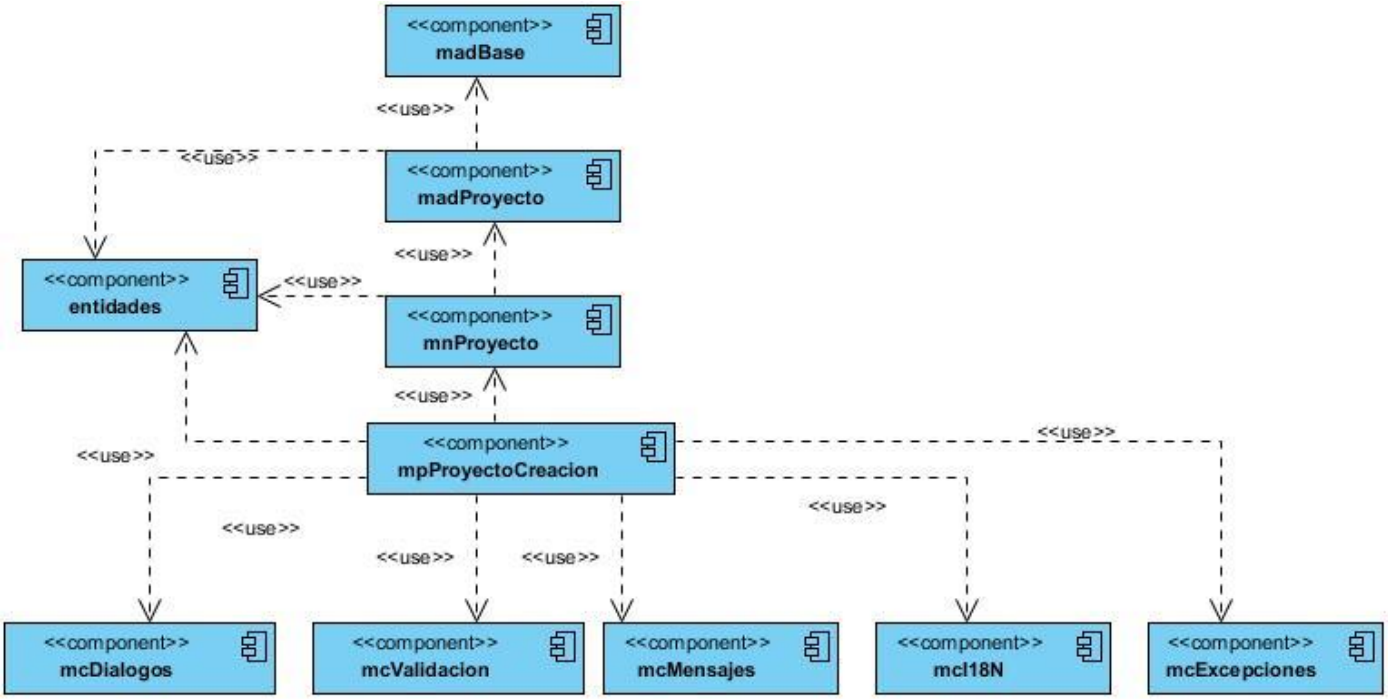

Figura 20 Diagrama de componentes Crear Proyecto.

3.3.2 Estándares de codificación

Los estándares de codificación definidos para las actividades de implementación en el proyecto (54) contribuyen al entendimiento del equipo de trabajo, limpieza del código y actividades de mantenimiento. Las principales normas utilizadas son:

- Los nombres de los módulos de presentación deben ser nomenclados empezando con “mp”. Ejemplo “mpCertificadosDigitales”.
- Los nombres de los módulos del negocio deben ser nomenclados empezando con “mn”. Ejemplo “mnCertificadosDigitales”.
- Los nombres de los módulos de acceso a datos deben ser nomenclados empezando con “mad”. Ejemplo “madCertificadosDigitales”.
- Los nombres de las interfaces de gestión del negocio deben ser nomenclados comenzando con “GN”. Ejemplo GNCertificadosDigitales.
- Los nombres de las interfaces gestoras de acceso a datos deben ser nomenclados comenzando con “GAD”. Ejemplo “GADCertificadosDigitales”.

- Los nombres de las clases que implementan a las interfaces terminan en “Imp”. Ejemplo “GADCertificadosDigitalesImp”.
- Todos los ficheros fuente deben comenzar con un comentario en el que se lista el nombre de la clase, información de la versión y fecha. Ejemplo

  “/*

  *Nombre de la clase

  *

  *Información de la versión

  *

  *Fecha

  */”

3.4 Verificación y validación de la solución

La verificación de la solución permite comprobar que el sistema se ha construido correctamente a partir de la validación de las actividades de la metodología. Para garantizar esto se verificaron las etapas de requisitos, diseño e implementación a partir del uso de técnicas existentes en la ingeniería de software.

3.4.1 Verificación de requisitos

El resultado del trabajo realizado es una consecuencia de la ingeniería de requisitos (especificación del sistema e información relacionada) y es evaluada su calidad en la fase de validación. La validación de requisitos examina las especificaciones para asegurar que todos los requisitos del sistema han sido establecidos sin ambigüedad, sin inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos, y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto (43).

Durante esta etapa se desarrollaron un número de tareas, las que se explican a continuación:

Validación de requisitos mediante prototipos: se presentaron los prototipos elaborados durante la especificación de requisitos a grupos especializados en los procesos, a fin de validar si el análisis realizado responde a las necesidades y aspiraciones del cliente. Los prototipos están descritos en la Especificación de casos de uso del sistema (49) del expediente del proyecto.

Revisión técnica formal de requisitos: se realizan revisiones de los requisitos obtenidos por parte del equipo de calidad del proyecto y del centro, se corrigen las no conformidades identificadas hasta ser liberada la documentación.

Liberación de requisitos: los artefactos de Especificación de Requisitos de Software y Especificación de Casos de uso son revisados por el equipo de calidad del centro CEGEL y Calisoft, generando un Acta de Aceptación en cada caso, luego de ser corregidas las no conformidades identificadas.

Aplicación de métricas: durante el desarrollo de la aplicación se aplicaron las métricas Especificidad de requisitos, Estabilidad de los requisitos y Grado de validación de los requisitos.

Especificidad de los requisitos: el objetivo de esta métrica es cuantificar la especificidad o falta de ambigüedad en la definición de los requerimientos. Para calcular esta métrica deben contarse los requerimientos que tuvieron igual interpretación por los revisores y compararlos con el total de requerimientos definidos. La especificidad de los requisitos se calcula como:

$$Q_1 = \frac{n_{ui}}{n_r}$$

Ecuación 1 Especificidad de los requisitos.

Donde Q1 representa la especificidad de los requisitos a calcular, Nr al total de requisitos definidos y Nui al total de requisitos para los que los revisores tuvieron interpretaciones idénticas.

$$Q_1 = {}^{21}/_{21} = 1$$

El valor de esta métrica debe estar siempre entre 0 y 1. Mientras más cerca de 1 esté el valor de ER mayor será la consistencia de la interpretación de los revisores para cada requerimiento y menor será la ambigüedad en la especificación de los requerimientos.

Estabilidad de los requisitos: el objetivo de esta métrica es medir cuan estables son los requisitos para asegurar su adecuación antes de pasar a la próxima disciplina. Se considera que los requerimientos son estables cuando no existen adiciones o supresiones en ellos que impliquen modificaciones en las funcionalidades principales de la aplicación. La estabilidad de los requisitos se calcula como:

$$ETR = \left[\frac{RT - RM}{RT}\right] * 100$$

Ecuación 2 Estabilidad de los requisitos.

Donde ETR representa la estabilidad de los requisitos a calcular, RT al total de requisitos definidos y RM a la cantidad de requisitos modificados.

$$ETR = \left[{}^{(21-0)}/_{21}\right] * 100 = 100$$

Esta métrica ofrece valores entre 0 y 100. El mejor valor de ETR es el más cercano a 100 porque mostrará que no se han realizado cambios sobre los requisitos, estos son estables y, por tanto, es confiable trabajar el análisis y diseño sobre ellos.

Grado de validación de los requisitos: los requisitos deben ser posibles de validar. La validación de los requisitos se realiza en consenso con el equipo de desarrollo al contrastar lo que desea el cliente con la posibilidad real de implementarlo. El grado de validación de los requerimientos mide la corrección en la definición de los requerimientos. Este valor se calcula como:

$$Q_3 = \frac{n_c}{(n_c + n_{nv})}$$

Ecuación 3 Grado de validación de los requisitos.

Donde Q3 representa el grado de validación de los requisitos a calcular, Nc al total de requisitos validados correctamente y Nnv al total de requisitos no validados.

$$Q_{3=} {}^{21}/_{(21+0)} = 1$$

El resultado de esta métrica está entre 0 y 1. El valor óptimo, es el más cercano a 1 e indica un alto nivel de corrección en la definición de los requerimientos.

3.4.2 Verificación de diseño

Métrica 1: Tamaño Operacional de Clase (TOC)

Para evaluar el diseño de la solución, se decide aplicar la métrica TOC a 92 clases del Diagrama de clases del diseño y se obtuvo un promedio de procedimientos de 4.663043478, llegando a la conclusión de que aproximadamente el 72% de las clases analizadas tienen poca responsabilidad (Figura 21), esta característica provoca que la mayoría de las clases sean reutilizables (Figura 23), se obtuvo además un 72% de complejidad baja (Figura 22), lo que significa que la mayoría de las clases no son muy complejas y posibilita una implementación más sencilla.

Figura 21 Resultados de la métrica TOC.

Figura 22 Resultados de la métrica TOC.

.

Figura 23 Resultados de la métrica TOC.

Métrica 2: Relaciones entre clases (RC)

La métrica RC se aplica igualmente a 92 clases del Diagrama de clases del diseño con un promedio de asociaciones de uso de 0,90217 llegando a la conclusión de que en las clases predomina un bajo acoplamiento (Figura 24), lo que significa una escasa dependencia de una clase respecto a otra y posibilita un mejor entendimiento del código, se obtuvo también una complejidad de mantenimiento baja (Figura 25), garantizando con esto que se necesite un grado de esfuerzo bajo para arreglar o modificar el código de la aplicación, así como una cantidad de pruebas baja (Figura 26) y reutilización alta (Figura 27) de las clases.

Figura 24 Resultados de la métrica RC.

Figura 25 Resultados de la métrica RC.

Figura 26 Resultados de la métrica RC.

Figura 27 Resultados de la métrica RC.

3.4.3 Verificación de implementación

La estrategia de verificación de la implementación se trazó a partir de la aplicación de pruebas de caja blanca y pruebas de caja negra. A continuación se muestran los resultados obtenidos.

Pruebas de caja blanca

En ocasiones llamada prueba de caja de cristal, es un método de diseño que usa la estructura de control descrita como parte del diseño al nivel de componentes para derivar los casos de prueba. Al emplear los métodos de prueba de caja blanca, el ingeniero del software podrá derivar casos de prueba que garanticen que todas las rutas independientes dentro del módulo se han ejercitado por lo menos una vez, ejerciten los lados verdadero y falso de todas las decisiones lógicas, ejecuten todos los bucles en sus límites y dentro de sus límites operacionales, y ejerciten estructuras de datos internos para asegurar su validez (43).

Para realizar la prueba es necesario efectuar primeramente el análisis de complejidad del algoritmo sobre el que se va a realizar la prueba, con el propósito de calcular los valores de la complejidad ciclomática. En la Figura 28 se observa un ejemplo de uno de los métodos al cual se le aplicó esta prueba.

```
private void habilitarCertificado() {
    IMensaje gestorMensajes = Lookup.getDefault().lookup(IMensaje.class);      //1
    I18N gestorIdioma = Lookup.getDefault().lookup(I18N.class);                //2
 if(this.pnlGestionarCertificadosDigitales1.tblCertificados.getObjetosChecked().isEmpty()) { //3
      String mensaje = gestorIdioma.getTexto(gestionarCertificadoDigitalTopComponent.class, "MA_SeleccionHabilitar"); //4
      gestorMensajes.notificarAdvertencia(mensaje);   //5
          }
       else {
 if(gestorMensajes.notificarConfirmacion(gestorIdioma.getTexto(gestionarCertificadoDigitalTopComponent.class, "MC_Habilitar"))
    == gestorMensajes.CONF_SI)//6
  {
  for (Iterator it = this.pnlGestionarCertificadosDigitales1.tblCertificados.getObjetosChecked().iterator(); it.hasNext();)//7
       {
        CertificadoDigital cert = (CertificadoDigital) it.next();   //8
        cert.setActivo(true);                                       //9
        this.pnlGestionarCertificadosDigitales1.updateUI();         //10
       }
        cambioRealizado = true;                                      //11
        }
      }
   }    //12
```

Figura 28 Código fuente de la funcionalidad habilitarCertificado().

Luego de este paso, es necesario representar el grafo de flujo asociado (Figura 29), en el cual se representan distintos componentes como es el caso de:

Nodo: son los círculos representados en el grafo de flujo, el cual representa una o más secuencias del procedimiento, donde un nodo corresponde a una secuencia de procesos o a una sentencia de decisión. Los nodos que no están asociados se utilizan al inicio y final del grafo.

Nodo predicado: son los nodos que contienen una condición y se caracterizan porque de ellos salen dos o más aristas.

Aristas: son constituidas por las flechas del grafo, son iguales a las representadas en un diagrama de flujo y constituyen el flujo de control del procedimiento. Las aristas terminan en un nodo, inclusive cuando el nodo no representa la sentencia de un procedimiento.

Regiones: son las áreas delimitadas por las aristas y nodos donde se incluye el área exterior del grafo, como una región más. Las regiones se enumeran, siendo la cantidad de regiones equivalente a la cantidad de caminos independientes del conjunto básico de un procedimiento.

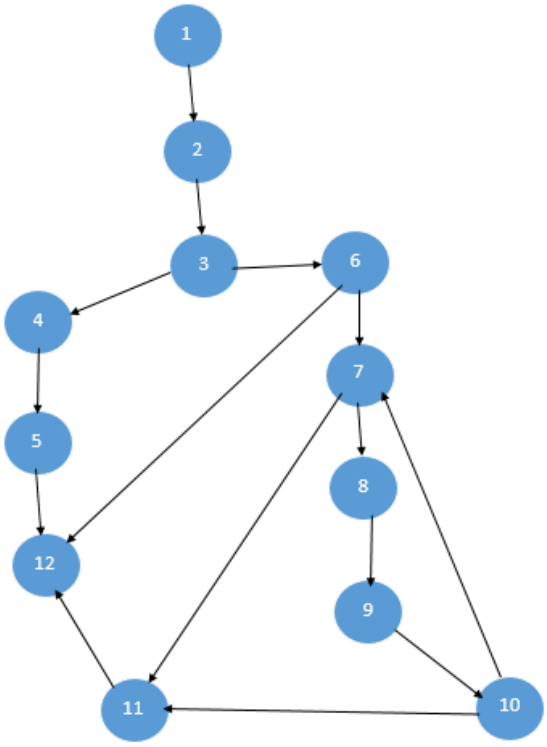

Figura 29 Grafo del flujo asociado a la funcionalidad habilitarCertificado().

Finalizada la construcción del grafo se realiza el cálculo de la complejidad ciclomática mediante tres fórmulas descritas a continuación, las cuales deben dar un mismo resultado para asegurar que el cálculo de la complejidad es correcto.

1. $\mathbf{V(G) = (A - N) + 2}$
Siendo "A" la cantidad total de aristas y "N" la cantidad total de nodos.

$V(G) = (15 - 12) + 2 \ = 5$

2. $\mathbf{V(G) = P + 1}$

Siendo "P" la cantidad total de nodos predicados (son los nodos de los cuales parten dos o más aristas).

$\mathrm{V(G) = 4 + 1 = 5}$

3. $\mathbf{V(G) = R}$

Siendo "R" la cantidad total de regiones, para cada fórmula "V (G)" representa el valor del cálculo.
$\mathrm{V(G) = 5}$

El cálculo efectuado utilizando las fórmulas antes descritas han dado como resultado el mismo valor en todos los casos, su resultado fue 5, lo que indica que existen 5 posibles caminos independientes por donde el flujo puede circular, y determina la cantidad de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. Seguidamente se representan los caminos básicos por los que puede recorrer el flujo:

Camino básico #1: 1-2-3-6-12

Camino básico #2: 1-2-3-4-5-12

Camino básico #3: 1-2-3-6-7-11-12

Camino básico #4: 1-2-3-6-7-8-9-10-11-12

Camino básico #5: 1-2-3-6-7-8-9-10-7-11-12

Se ejecutan pruebas para cada camino básico para verificar si se obtienen los resultados esperados con la ejecución del código. Por ejemplo, para probar el camino básico #2 se establecen las siguientes condiciones:

Condición de ejecución	Se necesita que el usuario seleccione la opción habilitar
Entrada	Al menos un certificado digital
Resultados esperados	Certificados digitales habilitados

Como parte de las pruebas de caja blanca aplicadas se hizo empleo de la herramienta JUnit 3, cuya utilización permitió evidenciar una ejecución correcta del código.

Resultado de las pruebas

Como resultado de la realización de las pruebas de caja blanca se probaron 18 de las funcionalidades más relevantes, de ellas se lograron optimizar 7 y se eliminaron errores de 4 de las mismas.

Pruebas de caja negra

También denominadas pruebas de comportamiento, se concentran en los requisitos funcionales del software. Es decir, permiten al ingeniero de software derivar conjuntos de condiciones de entrada que ejercitarán por completo todos los requisitos funcionales de un programa. La prueba de caja negra no es una opción frente a las técnicas de caja blanca. Es, en cambio, un enfoque complementario que tiene

probabilidades de descubrir una clase diferente de errores de los que se descubrían con los métodos de caja blanca. Las pruebas de caja negra tratan de encontrar errores en las siguientes categorías: 1) funciones incorrectas o faltantes, 2) errores de interfaz, 3) errores de estructuras de datos o en acceso a bases de datos externas, 4) errores de comportamiento o desempeño, y 5) errores de inicialización y término.

A diferencia de las pruebas de caja blanca, que se realizan al inicio del proceso de prueba, las de caja negra tienden a aplicarse durante las últimas etapas de la prueba. Debido a que estas desatienden a propósito la estructura de control, la atención se concentra en el dominio de la información (43). Para la validación del software se elaboraron 13 casos de pruebas (55) (56) (57) (58) (59) (60) (61) (62) (63) (64) (65) (66) (67) las cuales están descritas en los documentos de casos de prueba del proyecto.

Resultado de las pruebas

Se realizaron 3 iteraciones revisando en cada una la aplicación contra los 13 casos de pruebas elaborados. Durante la primera iteración se obtuvo un total de 15 no conformidades, de las cuales 9 de complejidad baja y 6 de complejidad media, durante la segunda iteración se encontraron 3 no conformidades, 2 de complejidad media y 1 de complejidad baja.

	Primera iteración	**Segunda iteración**	**Tercera iteración**
Casos de uso	13	13	13
No conformidades	15	3	0

3.4.4 Validación

Se define trazabilidad como la asociación del requisito con otros requisitos y las diferentes instancias con que se relaciona durante la evolución de las diferentes fases del ciclo de desarrollo del software. Esa asociación se controla en ambos sentidos, de los requisitos a los resultados y viceversa. La intención principal es poder determinar si todos los requisitos base han sido considerados y si las instancias que han sido generadas pueden asociarse con un requisito válido.

La Matriz de Trazabilidad de Requisitos ayuda a realizar seguimiento a los requisitos a lo largo del ciclo de vida del software para asegurar que se están cumpliendo de manera eficaz. Constituye un apoyo para el desarrollo del software ya que permite controlar el flujo de requisitos hasta la conclusión del desarrollo.

La matriz de trazabilidad realizada para la investigación presenta la trazabilidad de los casos de prueba (por las filas) a los casos de uso (por las columnas) desarrollados, mostrando como cada uno de los casos de

prueba se han hecho corresponder con uno de los casos de uso. Esto evidencia que cada uno de los requisitos acordados con el centro CEGEL fue satisfecho en su totalidad a partir de la implementación de los casos de uso, ya que los mismos muestran relación directa además con las clases de diseño, los casos de pruebas e interfaz de usuario.

CP\CU	Configurar apariencia	Configurar módulos	Crear proyecto	Generar producto	Gestionar idioma de	Gestionar idioma del producto	Gestionar marcos de apariencia	Gestionar módulos	Gestionar proyecto	Gestionar temas de apariencia	Seleccionar idioma	Configuraciones adicionales
CP_Crear proyecto			X									
CP_Configurar módulos		X										
CP_Gestionar idiomas del producto						X						
CP_Configurar apariencia	X											
CP_Gestionar idiomas de plataforma					X							
CP_Generar producto				X								
CP_Gestionar temas de apariencia										X		
CP_Gestionar proyecto									X			
CP_Configuraciones adicionales												X
CP_Gestionar módulos								X				
CP_Seleccionar idioma											X	
CP_Gestionar marcos de apariencia							X					

Figura 30 Matriz de trazabilidad.

3.4.5 Aceptación de la solución y resultados introducidos

Los resultados obtenidos a partir de la realización de esta investigación son:

- Acta de Liberación de requisitos de software Subsistema Plataforma.
- Acta de Liberación de requisitos no funcionales Subsistema Plataforma.
- Aceptación del Centro CEGEL de la solución de software.
- Aceptación del Proyecto DigiPRO de la solución de software.

- Aceptación del Grupo de Investigación de Informática Jurídica de la solución de software.

3.5 Conclusiones parciales

La realización de esta etapa permitió obtener la implementación de la plataforma para la construcción de sistemas de digitalización de documentos a partir de la ejecución de las tareas de ingeniería. La aplicación de pruebas unitarias permitió verificar que la aplicación obtenida satisface los requisitos especificados. En la verificación del diseño se analizaron las clases de diseño que juegan un papel fundamental en los procesos principales del sistema, a las que se les aplicó las métricas Tamaño Operacional de Clases y Relaciones Entre Clases, obteniendo resultados satisfactorios en este sentido.

Conclusiones

Al concluir el presente trabajo se puede afirmar que:

1. Los sistemas DigiPyrus, DigiDAP y CDA constituyen aportes significativos a las funcionalidades y características arquitectónicas que debe estandarizar la plataforma DigiPRO para la construcción de sistemas para la generación de objetos digitales con valor legal.
2. El uso de las métricas Tamaño Operacional de Clases y Relaciones Entre Clases arrojó como resultado que la solución propuesta es fácil de mantener y posibilita una alta reutilización de sus componentes.
3. Las pruebas de caja blanca permitieron verificar la correctitud de la implementación, y a partir de la solución de las no conformidades se obtuvo un sistema estable que posibilitó la realización de otro método de prueba, caja negra.
4. Con el diseño e implementación del subsistema Plataforma del proyecto DigiPRO se lograron satisfacer los requisitos acordados con la institución para la creación y configuración de sistemas para la obtención de objetos digitales con valor legal.
5. Se logró estandarizar la construcción de sistemas para la generación de objetos digitales con valor legal.

Recomendaciones

A partir de la investigación realizada se recomienda:

- ✓ Realizar investigaciones con versiones más avanzadas de NetBeans para determinar si es posible realizar mejoras a la internacionalización de la Plataforma.

Bibliografía

1. *Actas de la Conferencia General.* Colectivo de autores. París : Organización de las Naciones Unidas para la Educación , la Ciencia y la Cultura, 2003. Vol. 1. 1.

2. *Declaración de Vancouver: La Memoria del Mundo en la era digital: digitalización y preservación.* Vancouver, Canadá : s.n., 2012.

3. *Library digitization projects, issues and guidelines: A survey of the literature.* Lopatin, Laurie. 2, New York : Library Hi Tech, 2006, Vol. 24, págs. 273-289.

4. Portal de Archivos Espeñoles. [En línea] http://pares.mcu.es/.

5. FOGA. [En línea] http://www.foga.es/detalle_noticia.php?noticia=57&titlulo=&PHPSESSID=8ce4528da75dee425683a17c791bcb7f.

6. XEROX. [En línea] http://www.xerox.com/.

7. HP. [En línea] http://www8.hp.com/es/es/home.html.

8. Kodak. [En línea] http://www.kodak.com/ek/US/en/Home.htm.

9. *Plataforma para la digitalización de documentos DigiPRO: un proyecto de investigación y desarrollo resultante del binomio Universidad-Empresa.* Lizama, Yadira. La Habana : s.n., 2013. XIII Encuentro TECNOGEST .

10. Jacobson, Ivar, Booch, Grady y Rumbaugh, James. *El Proceso Unificado de Desarrollo de Software.* 2000.

11. *Desarrollo de Software Basado en Componentes.* Montilva C., Jonás A., Arapé, Nelson y Colmenares, Juan Andrés. Mérida : IV Congreso de Automatización y Control CAO03, 2003.

12. Sametinger, J. *Software engineering with reusable components.* s.l. : Springer Verlag, 1997.

13. Jag, Sodhi y Sodhi, P. *Software reuse: Domain analysis and design process.* s.l. : McGraw-Hill, 1999.

14. Ian, Sommerville. *Sotfware engineering.* 6. s.l. : Addison-Wesley, 2000.

15. *Technical concepts of component-based software engineering.* F. Bachmann, L. Bass, Ch. Buhman, S. Comella-Dorda, F. Long, J. Robert, R. Seacord, K. Wallnau. s.l. : Carnegie Mellon University, 2000, Vol. II.

16. Red de aprendizaje. [En línea] http://www.reddeaprendizaje.com/inicio/item/47-plataforma-informatica.

17. Llorente, Alvaro. Plataforma de hardware completa. [En línea] 2011. http://www.industriaembebidahoy.com/plataforma-de-hardware-completa/.

18. ARDUINO. [En línea] http://arduino.cc/en/Main/Products?from=Main.Hardware.

19. Xbox. Xbox. [En línea] http://www.xbox.com/es-es/#fbid=S5K2y6H_bss.

20. Sony. PlayStation. [En línea] http://es.playstation.com/.

21. *The Java EE 6 Tutorial Basic Concepts.* s.l. : Addison.Wesley, Agosto de 2010, Vol. IV.

22. Oracle. Oracle. [En línea] http://www.oracle.com/es/index.html.

23. Interempresas. [En línea] http://www.interempresas.net/ObrasPublicas/FeriaVirtual/Producto-Plataforma-de-hardware-y-software-Flexible-Machine-67420.html.

24. *¿QUÉ ES EL CLOUD COMPUTING? Recomendaciones para Empresas.* del Río, Mariano M. s.l. : Programa Nacional de Infraestructuras Críticas de Información y Seguridad.

25. DIGIBÍS. [En línea] [Citado el: 12 de Diciembre de 2013.] http://www.digibis.com.

26. DIGIARCH. [En línea] http://www.digibis.com/software/digiarch.html.

27. MICROFILE. [En línea] 2013. http://www.microfile.com.uy/.

28. XEROX. *XEROX.* [En línea] http://www.xerox.com/news/news-archive/esar.html.

29. La Rosa Montes, Dayana. *Análisis y Modelado del sistema para el proceso de digitalización de documentos de Registros y Notarías del MPPRIJ de la República Bolivariana de Venezuela.* 2007.

30. *Software para la Digitalización y Gestión Documental de los Registros y Notarías del MIJ de Venezuela.* Martori Domenech, Yunior y Queizán Pérez, René. 2007.

31. González Valdés, Sandra y Suárez Font, René. *Diseño e Implementación de los módulos de Preparación de Documentos, Digitalización de Documentos y Asociación de Metadatos del Centro de Digitalización para la División de Antecedentes Penales.* 2011.

32. Hernández Toledo, Alejandro y Borges Piedra, Liliana. *Diseño e implementación de los módulos de Recepción y Devolución, Control de Calidad y Otorgamiento de Documentos del Centro de Digitalización para la División de Antecedentes Penales.* 2011.

33. *Modelo de Referencia para la Digitalización de Fondos Documentales de Antecedentes Penales.* Lizama, Yadira. La Habana : s.n., 2011.

34. Lizama, Yadira. *Proyecto Técnico DigiPRO.* La Habana : Centro de Gobierno Electrónico, 2012.

35. Booch, Grady, Rumbaugh, James y Jacobson, Ivar. *El Lenguaje Unificado de Modelado.* s.l. : ADDISSON WESLEY, 2007.

36. Visual Paradigm. *Visual Paradigm.* [En línea] [Citado el: 29 de Diciembre de 2014.] http://www.visual-paradigm.com/product/vpuml/.

37. *Using Rose.*

38. *A Swing architecture overview.* Fowler, Amy. s.l. : java.sun.com, 1998.

39. Petri, Jürgen. *NetBeans Platform 6.9 Developer's Guide.* Birmingham : Packt Publishing Ltd, 2010.

40. JUnit. *JUnit.* [En línea] JUnit.org.

41. Sommerville, Ian. *Software Engineering.* 8. s.l. : Addison-Wesley.

42. Padrón , Aylin, Gasca, Greicy y Lizama, Yadira. *Especificación de requisitos de software. Subsistema Plataforma.* La Habana : Centro de Gobierno Electrónico, 2012.

43. Pressman, Roger S. *Ingeniería de Software. Un enfoque práctico.* 6. s.l. : McGraw'Hill Companies, 2007.

44. Lizama, Yadira, Padrón, Ailyn y Gasca, Greicy. *Especificación de requisitos no funcionales. Subsistema Plataforma.* La Habana : Centro de Gobierno Electrónico, 2012.

45. Moré, Dailien. *Estándares de Diseño de Interfaces de Usuario.* La Habana : Centro de Gobierno Electrónico, 2012.

46. Concepción, Mario. *Estándares de Diseño.* La Habana : Centro de Gobierno Electrónico, 2012.

47. Edghill, Yanet. *Estándares para la Especificación de Requisitos.* La Habana : Centro de Gobierno Electrónico, 2012.

48. Booch, Grady, Rumbaugh, James and Jacobson, Ivar. *The Unified Modeling Language User Guide.* Massachusetts : Addison-Wesley Longman Inc, 1998. 1998. 0-201-57168-4.

49. Padrón, Ailyn y Gasca, Greicy. *Especificación de casos de uso. Subsistema Plataforma.* La Habana : Centro de Gobierno Electrónico, 2012.

50. Larman, C. *UML y Patrones. Introducción al Análisis y Diseño Orientado a Objetos.* México : s.n., 1999.

51. [En línea] www.scribd.com.

52. Soto, Jesús y Cebrián, Fidel. *Modelo de diseño.* La Habana : Centro de Gobierno Electrónico, 2014.

53. Larman, Craig. *UML y Patrones.* 2003.

54. Fadraga, Lissuan. *Estándares de Codificación para Java.* La Habana : Centro de Gobierno Electrónico, 2012.

55. Cebrián, Fidel y Soto, Jesús. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Configuraciones adicionales.* La Habana : Centro de Gobierno Electrónico, 2014.

56. Cebrián , Fidel y Soto, Jesús. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Configurar apariencia.* La Habana : Centro de Gobierno Electrónico, 2014.

57. Cebrián, Fidel y Soto, Jesús. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Configurar módulos.* La Habana : Centro de Gobierno Electrónico, 2014.

58. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Crear proyecto.* La Habana : Centro de Gobierno Electrónico, 2014.

59. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Generar Producto.* La Habana : Centro de Gobierno Electrónico, 2014.

60. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestión de certificados digitales.* La Habana : Centro de Gobierno Electrónico, 2014.

61. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestión de idiomas de la plataforma.* La Habana : Centro de Gobierno Electrónico, 2014.

62. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestión de idiomas del producto.* La Habana : Centro de Gobierno Electrónico, 2014.

63. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestión de marcos de apariencia.* La Habana : Centro de Gobierno Electrónico, 2014.

64. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestión de módulos.* La Habana : Centro de Gobierno Electrónico, 2014.

65. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestión de temas de apariencia.* La Habana : Centro de Gobierno Electrónico, 2014.

66. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestionar proyecto.* La Habana : Centro de Gobierno Electrónico, 2014.

67. —. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Seleccionar idiomas.* La Habana : Centro de Gobierno Electrónico, 2014.

68. Flanaga, David. *Java en pocas palabras.* México : McGraw-Hill, 1998. 970-10-2070-7.

69. DIGIMUS. [En línea] http://www.digibis.com/software/digimus.html.

70. DDIGIHUB. [En línea] http://www.digibis.com/software/digihub.html.

71. DIGIOAI. [En línea] http://www.digibis.com/software/digioai-repositorio-oai-pmh.html.

72. Benk, Kent. *Extreme Programming Explained.* s.l. : Addison Wesley, 1999.

73. *Introduction to OpenUP (Open unified process).* Balduino, Ricardo. 2007, Eclipse site.

74. *Agile Unified Process.* Edeki, Charles. 3, s.l. : IJCSMA, 2013, International Journal of Compu, Vol. 1.

75. Cebrián, Fidel y Soto, Jesús. *Diseño de Casos de Prueba basado en Casos de Uso v2.0 Gestión de idiomas del producto.* La Habana : Centro de Gobierno Electrónico, 2014.