Universidad de las Ciencias Informáticas

Facultad 2

Título: Personalización del Xabal Arkheia para la Dirección General de Producción

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

Autora: Zurelis Nicolas Escalona

Tutor(es): Ing. Leandro Roura Sixto
Ing. Adys Aragón Rodríguez

La Habana, junio de 2015
"Año 57 de la Revolución"

"Este es tiempo virtuoso y hay que fundirse en él…"

José Martí

Declaración de Autoría

Declaro ser el único autor de este Trabajo de Diploma y autorizo al Centro de Informatización de la Gestión Documental de la Universidad de las Ciencias Informáticas; así como a dicha universidad para hacer uso del mismo en su beneficio y lo que estime pertinente.

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

Zurelis Nicolas Escalona

_____

Firma del Autor

Ing. Leandro Roura Sixto

_____

Firma del Tutor

Ing. Adys Aragón Rodríguez

_____

Firma de Tutor

Agradecimientos

A mi madre por todo su AMOR, apoyo y confianza.

A mi tutor, AMIGO y guía en este proyecto: Leandro.

A mi familia por siempre estar presente cuando lo necesité.

A mi hermana Paula por toda su ayuda para superar mis tropiezos y confiarme a su Princesa.

A mis amigos de cerca y de lejos, de antes y después, que confiaron en mí a pesar de mis errores y que siempre estuvieron presente apoyándome y con sus expectativas puestas en mí.

A mí, por ser fuerte y tenaz en mi pensamiento de continuar mis estudios.

Zurelis

Dedicatoria

A MI MADRE: Martha Beatriz Escalona Labaut, por ser la mejor.

Resumen

Actualmente la Dirección General de Producción en la Universidad de las Ciencias Informáticas que tiene la misión de organizar la producción, no cuenta con un sistema que le permita salvaguardar los archivos históricos de los procesos que se generan durante la creación de un software. Con el objetivo de darle solución a esta problemática se propone la personalización del Xabal Arkheia para la Dirección General de Producción.

Con la solución propuesta se logró una aplicación para salvaguardar los productos software creados en la universidad; dígase toda la documentación dispuesta como bibliografía de consulta, código fuente del software, y las tareas y usuarios asociados a la gestión de los proyectos. Para lograrlo se dispuso de funcionalidades que conforman los módulos del Sistema Xabal Arkheia, la creación de otras nuevas y la integración de dicho sistema con los sistemas Gespro y eXcriba, de donde se obtienen datos de los proyectos. Esta propuesta constituye una nueva forma de organización y control de las producciones en la universidad.

En el presente trabajo se realizó el análisis, diseño e implementación de nuevas funcionalidades para lograr la personalización, a partir de los requisitos de software definidos y la arquitectura establecida. Se realizó un análisis de conceptos y soluciones que resuelven problemas similares; y se adoptó la metodología, tecnologías, herramientas y lenguajes establecidos en la versión actual de Arkheia. Para comprobar el cumplimiento de los requisitos y el correcto funcionamiento del sistema, se aplicaron pruebas de calidad que permitieron eliminar la aparición de no conformidades.

Palabras clave: Arkheia, personalización, proyectos, archivos, ficheros, integración.

Índice

Índice

Introducción

La Dirección General de Producción (DGP) en la Universidad de las Ciencias Informáticas (UCI) tiene la misión de organizar la producción, estandarizando políticas de gestión de las oportunidades de proyectos, de la ejecución de proyectos y la infraestructura de desarrollo; implantando plataformas de servicios tecnológicos y el modelo de desarrollo tecnológico de la producción basada en la reutilización de activos de software y las arquitecturas de referencia de soluciones informáticas (1).

Actualmente la DGP no tiene un control centrado de todos los procesos que genera la trayectoria de vida de un proyecto, por lo que requiere de un sistema que le permita salvaguardar los archivos históricos. Por dicha razón se ha iniciado un proceso de recopilación y archivo de la memoria histórica del desarrollo de software en la UCI. Este proceso inicia cuando el sistema gestiona la creación de los cuadros de clasificación y la estructura de ubicación física de la institución; luego se desarrolla la etapa de procesamiento donde se realiza la descripción del proyecto a través de datos particulares, tales como: nombre oficial, número de registro, identificador, cliente, autores, objetivo general, alcance, entidad desarrolladora, fecha de inicio y fin, entre otros criterios necesarios en la descripción del documento software.

El Centro de Informatización de la Gestión Documental (CIGED) posee dentro de su cartera comercial de productos el sistema Xabal Arkheia[1], una aplicación informática responsable de la gestión de los documentos en archivos históricos, al tiempo que facilita su incorporación, su descripción -mediante la Norma Internacional General de Descripción Archivística ISAD (G) por sus siglas en inglés- y su conservación. El sistema permite, también, la creación del cuadro de clasificación de la institución y registrar el almacenamiento físico de los documentos. Debido a la misión del sistema de gestionar archivos históricos, la DGP tomó la decisión de utilizar el Arkheia para almacenar la información de todos sus proyectos terminados.

[1] Sistema de Gestión de Archivos Históricos.

Sin embargo, el sistema no cubre todas las necesidades de la DGP. Los metadatos de la Norma ISAD (G) en el Xabal Arkheia no se corresponden con los utilizados en el Xedro Gespro[2], sistema de donde se obtienen los datos de los proyectos realizados en la UCI.

Para almacenar el código y la documentación de un proyecto, se genera un archivo comprimido que puede sobrepasar los 500 megabytes (MB); valor que constituye el límite de tamaño para la adición de archivos en la versión actual de Arkheia. Producto a la constante adición de ficheros grandes al sistema se hace necesario monitorear el espacio en disco duro. Para ello se requiere de un panel de administración donde se visualice el estado de almacenamiento del servidor.

El cuadro de clasificación es la estructura lógica donde se ubican los documentos en un archivo de forma jerárquica o ramificada, puede ser lineal o no lineal. La versión actual de Arkheia permite configurar tantos cuadros como se desee pero de forma lineal, es decir, la forma de organización de diversos elementos de un determinado sistema en el que cada uno está subordinado al elemento inmediatamente superior (Un nivel determinado pertenece a un tipo de nivel y solo a ese, no puede pertenecer a otro). Esta funcionalidad no cumple con la estructura del cuadro de clasificación de la DGP, pues dicha Dirección requiere que un tipo de nivel pueda pertenecer a cualquier otro u otros.

Además requiere que luego de transcurridos dos años de introducido el documento en el archivo, el sistema notifique al usuario el tiempo del software en el Arkheia.

Esta situación problemática lleva a enunciar el siguiente **problema a resolver**: ¿Cómo almacenar la memoria histórica del desarrollo de software en la UCI?
El **objeto de estudio** está centrado en los procesos realizados en los archivos históricos.
El **campo de acción** de la investigación se enmarca en los procesos realizados en los archivos históricos de las instituciones productoras de software.
Para darle solución al problema planteado se propone como **objetivo general**: Personalizar el Xabal Arkheia para gestionar los procesos archivísticos de la DGP en la UCI.

Para dar cumplimiento al objetivo general se trazaron los siguientes **objetivos específicos**:

[2] Suite de Gestión de Proyectos.

1. Realizar el estudio del estado del arte y confeccionar el marco teórico de la investigación.
2. Realizar el análisis y diseño del Xabal Arkheia para la DGP.
3. Implementar el sistema y la integración con el eXcriba y el Gespro.
4. Realizar las pruebas de software.

Los objetivos específicos se desglosaron en las siguientes **tareas de investigación**:

1. Elaboración del marco teórico de la investigación para entender el funcionamiento y desarrollo de los archivos históricos.
2. Análisis de sistemas similares enmarcados en el objeto de estudio que permita la obtención de buenas prácticas para la personalización del Xabal Arkheia.
3. Descripción de las tecnologías, lenguajes y herramientas definidas para las personalizaciones del Xabal Arkheia.
4. Modelación de los procesos de gestión de archivos históricos en la DGP para conocer las características esenciales que debe poseer la personalización del Xabal Arkheia.
5. Especificación de los requisitos funcionales y no funcionales para conocer las características esenciales que requiere el sistema.
6. Descripción e implementación de los Casos de Usos para describir el funcionamiento del sistema.
7. Elaboración de casos de pruebas basados en casos de usos, los cuales sirven de guía para la aplicación de las pruebas al sistema.
8. Realización de pruebas de caja negra al producto para identificar y corregir posibles errores en el software.

Para el desarrollo del presente trabajo se utilizan diferentes métodos científicos, los cuales constituyen la forma de abordar la realidad, de estudiar los fenómenos de la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir la esencia de los mismos y sus relaciones (2).

Métodos Teóricos

- **Analítico-Sintético:** permitió realizar un análisis de la metodología, tecnologías, lenguajes y herramientas de desarrollo; identificando algunas de sus características distintivas y se ofrece una valoración de su uso en esta investigación.

- **Históricos - Lógicos:** permite una mejor comprensión del proceso archivístico, a partir del estudio de diversos sistemas de gestión de archivos mediante las normas establecidas.
- **Modelación:** permite crear abstracciones con el objetivo de explicar la realidad, en este caso se utiliza el Lenguaje Unificado de Modelado (UML) pues permite reflejar la estructura, comportamiento e interacción de la solución mediante el diseño de diagramas y además hacer representaciones de la posible interfaz de usuario del sistema.
- **Análisis documental:** se realiza un estudio de la bibliografía referente a las herramientas, metodología y tecnologías para el desarrollo del sistema, así como el estudio de procesos archivísticos para dar solución a la problemática.

Métodos Empíricos

- **Entrevista:** con el objetivo de obtener información referente al proceso que sigue la DGP para salvar su memoria histórica e identificar los elementos pertinentes para el diseño del sistema que requiere el cliente.

El presente documento consta de Introducción, cuatro capítulos donde se describe todo el proceso de la investigación, Conclusiones, Recomendaciones, Bibliografía, Referencias Bibliográficas, Anexos y Glosario de Términos.

Capítulo 1. Fundamentación Teórica: Aborda los fundamentos teóricos de la investigación a partir de la valoración de los conceptos fundamentales asociados al dominio del problema. Incluye conceptos del proceso archivístico; y se describen las tecnologías así como la metodología de desarrollo de software y herramientas utilizadas.

Capítulo 2. Características del sistema: Ofrece una descripción general de la propuesta de solución para la personalización del Xabal Arkheia para la DGP. Se define el modelo de negocio, se especifican los requisitos funcionales y no funcionales del sistema, se definen los actores que intervienen en los procesos y se ofrece una definición de los casos de uso a partir de su descripción y representación gráfica.

Capítulo 3. Análisis y diseño del sistema: En este capítulo se realiza el análisis y diseño del sistema, que proporciona una comprensión detallada de los requisitos e impone una estructura del sistema que contribuye a una arquitectura sólida que dará paso a la implementación. También se desarrollan los diagramas de estructura e interacción, y se define la arquitectura del software, el patrón de diseño y el modelo de datos.

Capítulo 4. Implementación y prueba: En este capítulo se desarrolla la implementación y prueba del sistema, a partir de los resultados del análisis y diseño. Como parte de la implementación se muestra, el diagrama de componentes y el diagrama de despliegue. Se definen las pruebas de software que garantizan la calidad y revisión final del cumplimiento de los requisitos y especificaciones de diseño.

Capítulo 1. Fundamentación Teórica

1.1 Introducción

En este primer capítulo se ofrece una descripción del marco teórico en el que se desarrolla la presente investigación, a partir de la valoración de conceptos fundamentales de los procesos archivísticos, conceptos que permitirán un mejor entendimiento de la problemática en cuestión. Además se describen las tecnologías así como la metodología de desarrollo de software, lenguajes y herramientas utilizadas.

1.2 Marco Conceptual

Dirección significa orientar los caminos para conseguir los objetivos de forma organizada, apoyándose en las tareas puntuales: planificar, organizar, controlar y gestionar (3). Y este es precisamente la misión principal de la Dirección General de la Producción en la UCI, organizar la producción: estandarizando políticas de gestión, ejecución y desarrollo de proyectos. Para lograrlo requiere de un sistema que permita el control y conservación de la memoria histórica de los procesos que genera la creación de un producto software en nuestra institución. Dígase documentación bibliográfica, código fuente, y tareas y usuarios asociadas al proceso de desarrollo del software. Toda esta información se obtiene de diversas fuentes a la vez independientes, por lo que se hace necesario la creación de un software que reúna todos estos datos en un solo espacio, el cual permita su acceso, consulta y preservación.

La investigación establece los términos fundamentales para un mejor entendimiento de los conceptos del negocio. A continuación se detallan algunos conceptos identificados y que resultan necesarios para la comprensión de dicha investigación:

Documento: Representación de la información sobre los objetos de la realidad objetiva y la actividad intelectual del hombre por medio de la escritura, la gráfica, la fotografía, la grabación u otro medio en cualquier portador (4).

Archivo: espacio que conserva los documentos de valor permanente (4).

Para la informática, un documento puede ser un texto, una imagen, un sonido, una animación o un video; el cual recibe el nombre de archivo informático en el momento que pasa a ser almacenado para su conservación. El archivo informático constituye el equivalente digital del archivo tradicional manuscrito, el

cual es identificado por un nombre y la descripción del directorio que lo contiene. Para esta investigación se considera como un archivo a un proyecto desarrollado en la UCI.

Unidad de conservación: es el espacio en donde se desarrolla la gestión, administración y ejecución de los diferentes procesos de conservación que se deben aplicar para la preservación en el tiempo de los documentos, sean éstos en soportes de papel, audiovisuales o electrónicos (5). En esta investigación la unidad de conservación representa la dirección física en el sistema fichero del almacenamiento del servidor.

Sistema de Archivo: es el conjunto de normas e instituciones que participan en la dirección, seguimiento, coordinación e inspección de los programas para la conservación, tratamiento y difusión del patrimonio documental. Lo componen: los archivos, los servicios archivísticos, la administración de archivos, la legislación archivística y el personal (4).

Cuadro de clasificación: esquema gráfico que muestra la clasificación y jerarquización dada a la documentación de un archivo. Instrumento técnico que refleja la estructuración de los grupos documentales, aporta los datos esenciales sobre dicha estructura, al contener divisiones clasificatorias por las cuales se distribuyen los documentos y la información contenida en ellos (4).

En la actualidad el número de aplicaciones informáticas ha crecido exponencialmente y cada una responde a una serie de necesidades específicas, por lo que se ha convertido en un problema para muchas instituciones la administración, el acceso a la información, la gestión de los costes, etc. En muchos casos, el disponer de numerosas aplicaciones informáticas se convierte en un problema más que en una solución. Muchas de estas aplicaciones son independientes entre sí, no se comunican, por lo que la información utilizada por cualquiera de ellos en un momento determinado no es compartida por el resto, y ni mucho menos tiene por qué ser la misma información con la que se está trabajando en las demás. Una posible solución para este problema es la **integración** con la cual se logra conectar aplicaciones o sistemas informáticos y compartir información, eliminar puntos de fallo, generar ahorros de costes, mejorar la eficiencia y llegar a conocer lo que realmente pasa para poder tomar decisiones más precisas y rápidas (6).

En esta investigación se requiere de la integración del sistema Xabal Arkheia, con el Gespro y el eXcriba, estos dos últimos son sistemas que brindan información de los proyectos, imprescindible para la descripción

general de los mismos. La integración de diferentes sistemas es necesaria para mejorar la administración de los datos.

Xedro Gespro: es una suite orientada a la web, que permite a las entidades cuyos productos deban ser planificados en forma de proyectos, el seguimiento y control de estos, ya sean proyectos de inversión, construcción, desarrollo de software o mantenimiento. La suite posibilita además la toma de decisiones a tres niveles: nivel de proyecto, nivel de entidad ejecutora y nivel gerencial (7).

Xabal eXcriba: Interfaz Web del Gestor de Documentos Administrativos. Es un software para la gestión documental, diseñado para tramitar los documentos administrativos que se generan o reciben dentro de las organizaciones a partir de sus funciones, por lo tanto, involucra todas las áreas de una organización, permitiéndoles gestionar de forma correcta la documentación como prueba, testimonio y evidencia de las actividades organizacionales (8).

1.3 Análisis de sistemas informáticos

A medida que se hace evidente el crecimiento en el tamaño de los fondos documentales se vuelve más compleja la gestión de los mismos, lo que da paso a la informatización de dichos documentos para lograr su recuperación, almacenamiento, control y preservación. La necesidad de cambiar obsoletos y agotadores procesos de trabajo en la gestión de archivos, conlleva al surgimiento de numerosas soluciones informáticas capaces de gestionar estos procesos de forma ágil, sencilla y segura. A continuación una muestra de algunos de estos sistemas que constituyen soluciones valiosas, creados dentro y fuera de nuestro país.

DOCUNET

Solución de gestión Documental (DOCUNET) es una aplicación informática centralizada bajo arquitectura web cuya funcionalidad está orientada a la edición de documentos estructurados y a su generación en fichero o en papel. Permite la administración de un centro documental y el control de préstamo de sus documentos físicos, adecuado para la conservación de la memoria institucional aplicando organización y normalización a su archivo, basado en los conceptos de gestión documental. Basada en la normatividad emitida por Archivo General de la Nación de Colombia (9). Se encuentra en la modalidad de Software Licenciado (10).

Archivist´s Toolkit

Archivists Toolkit, es un sistema de gestión de datos de archivos de código abierto para proporcionar un apoyo amplio e integrado para la gestión de archivos. Está dirigido a una amplia gama de depósitos de archivos. Sus principales objetivos son apoyar tratamiento archivístico y la producción de instrumentos de acceso, promover la normalización de datos, promover la eficiencia y reducir los costes de formación. Actualmente, la aplicación soporta acceso y descripción de los materiales de archivo, estableciendo nombres y temas asociados con el material de archivo (11).

SADE.Net

Sistema de Administración Documental, es un sistema de gestión empresarial para la administración integral e inteligente de toda la documentación de la empresa, cubriendo procesos como manejo de archivo, correspondencia, digitalización de documentos físicos, etc. Entre las características que el sistema posee se encuentran el control y la administración del archivo físico, transferencias documentales, préstamos, consulta de la documentación de una manera eficiente, rápida y segura, implementación de flujos de trabajo para automatizar procesos documentales, manejo de documentos electrónicos de cualquier formato. Se encuentra en la modalidad de Software Licenciado (10).

SIADOC

Solución para la Gestión Documental de Archivo y Correspondencia, es una herramienta para facilitar el control y la administración de los archivos y expedientes físicos. Compuesto por módulos que permiten la ejecución de las actividades y la administración del archivo de manera organizada. Proporciona un modelo de seguridad por niveles que le permite al usuario gran flexibilidad para controlar el acceso parcial o total del sistema. Se encuentra en la modalidad de Software Licenciado (10).

ArchivenHis

ArchivenHis es un sistema automatizado de gestión de documentos para un archivo histórico, fue creado en la UCI para el Archivo General de la República Bolivariana de Venezuela, solución que ha sido de interés para varias empresas tanto nacionales como internacionales debido a las soluciones que ofrece. Permite la conservación y difusión de la documentación existente de un archivo, y efectuar búsqueda de las versiones digitales que existan sobre los documentos. Sin embargo, no es personalizable a los requerimientos de nuevos clientes por estar desarrollado de acuerdo a las necesidades específicas del cliente venezolano.

ArchivenHIS no presenta funcionalidades de administración informática, solo presenta una gestión básica de usuarios con permisos ya definidos previamente y no gestionables. Además no posee una administración de parámetros generales de la aplicación como son nomencladores, valores de configuración y auditoría de las acciones de los usuarios.

Resultados del análisis de sistemas informáticos

La solución que se propone para esta investigación requiere de una aplicación especializada en el registro de los proyectos que se desarrollan en la UCI, que permita el manejo del documento software y su gestión archivística; pero también debe integrarse a otros dos sistemas creados en nuestra universidad de los cuales se van a obtener datos que precisa la descripción del archivo. Los sistemas analizados (DOCUNET, SADE.net, SIADOC, Archivists Toolkit) no aportan nuevas ideas para nuestra solución, pues los mismos están dedicados sólo a la gestión de documentos por lo que no responden al requisito fundamental del sistema sugerido. En el caso del ArchivenHis, se desarrolló bajo características concretas requeridas por el Archivo General de la Nación de la República de Bolivariana Venezuela para la gestión de documentos para un Archivo Histórico.

Por todo lo analizado anteriormente se concluye que el sistema Xabal Arkheia, es el adecuado para el desarrollo del sistema resultante de esta investigación, porque es fácil de personalizar y se tiene toda la documentación técnica de la implementación del sistema.

1.4 Metodología, herramientas, lenguajes y tecnologías

Durante el proceso de construcción de un software es importante definir las normas de desarrollo, para obtener con éxito el producto final, asegurando la calidad del mismo. Para el desarrollo de la personalización del Arkheia para la DGP se adoptó la metodología definida para dicho sistema en su versión actual, así como sus herramientas, lenguajes y tecnologías, lo que queda establecido en el documento Arquitectura Vista de entorno de CIGED – Arkheia para la Red de Centros (12). A continuación se ofrece una breve descripción de dichas normas definidas por el equipo de desarrollo del Arkheia.

1.4.1 Metodología de desarrollo

Proceso Unificado de Rational (RUP)

El Proceso Unificado de Desarrollo Software o simplemente Proceso Unificado es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. Posee tres características

clave que lo hace único: dirigido por casos de uso, centrado en la arquitectura, e iterativo e incremental. Rational Unified Process (RUP) constituye una especificación más detallada del Proceso Unificado, es un proceso de desarrollo de software que junto con el Lenguaje Unificado de Modelado (UML), es usada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. Esta metodología de desarrollo provee una amplia documentación, que es fundamental para las consultas y referencias (13).

1.4.2 Lenguajes de modelado y desarrollo

UML (Unified Modeling Language / Lenguaje unificado de modelado)

Es el lenguaje de modelado usado para visualizar, especificar, construir y documentar un sistema. Brinda un patrón para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación y esquemas de bases de datos. Puede ser usado independientemente de la metodología de desarrollo seleccionada y permite modelar el flujo de desarrollo mediante los diagramas de casos de uso, de clases, de actividades, entre otros (14).

BPMN o Notación para el Modelado de Procesos de Negocio

BPMN o Business Process Modeling Notation es una notación basada en diagramas de flujo para definir procesos de negocio, desde los más simples hasta los más complejos para dar soporte a la ejecución de procesos. En BPMN los procesos de negocio involucran la captura de una secuencia ordenada de las actividades e información de apoyo. Proporciona una notación estándar que es fácilmente entendible por parte de todos los involucrados e interesados del negocio, y tiene como fin servir como lenguaje común para unificar la etapa del diseño de los procesos de negocio y su implementación (15).

HTML (Hyper Text Markup Languaje / Lenguaje de Marcado de Hipertexto)

HTML es un lenguaje de composición de documentos y especificación de ligas de hipertexto que define la sintaxis y coloca instrucciones especiales que no muestra el navegador, aunque sí le indica cómo desplegar el contenido del documento, incluyendo texto, imágenes y otros medios soportados. También le indica cómo hacer un documento interactivo a través de ligas especiales de hipertexto, las cuales conectan diferentes documentos, así como otros recursos de Internet. Define la estructura y apariencia básica de documentos y conjuntos de documentos (16).

JavaScript

Es un lenguaje de programación para crear páginas web dinámicas, que son aquellas que incorporan efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Es un lenguaje ideal para agregar ciertas funciones rápidas a una página Web; muy potente pues solo hay que crear el código y cargarlo; gran usabilidad, lo que se evidencia en las miles de páginas Web publicadas, que incorporan elementos que lo usan (17).

Groovy

Groovy es un lenguaje dinámico y ágil, implementado sobre la plataforma Java, por lo que se integra completamente con librerías y código Java. Además, la mayoría del código Java es sintácticamente correcto en Groovy. Dicho lenguaje ha sido inspirado por lenguajes como Python, Ruby and Smalltalk, usando una sintaxis análoga a Java. Es el lenguaje en el que se escriben las aplicaciones en Grails, o al menos la mayor parte de los artefactos que contienen. (18) Este lenguaje simplifica la sintaxis de Java para expresar lo que queremos hacer, aumentando la productividad (19).

Java

Java es un lenguaje que tiene su propia estructura, reglas de sintaxis y paradigma de programación al igual que otros lenguajes. El paradigma de programación del lenguaje Java se basa en el concepto de programación orientada a objetos (POO), que usa los objetos en sus interacciones, para diseñar aplicaciones y programas informáticos. Su finalidad es crear aplicaciones de todo tipo (20).

1.4.3 Herramientas de modelado y desarrollo

Visual Paradigm for UML 8.0

Visual Paradigm es una herramienta que soporta el modelado mediante UML y proporciona asistencia al equipo de desarrollo durante el ciclo de vida de un Software. Su uso facilita el modelado de UML y controla que este sea correcto, hace posible la coherencia entre diagramas evitando duplicidades, se puede integrar con otras aplicaciones, permite el trabajo en grupo y la reutilización de modelos utilizados en otros proyectos. Posee excelente interoperabilidad con otras herramientas y la mayoría de los principales Entornos de desarrollo integrado (IDEs) (21).

SpringSource Tool Suite (STS) 2.5

Es un entorno de desarrollo integrado de código abierto desarrollado por SpringSource que soporta una amplia gama de lenguajes de programación. Posee soporte para integrar el contenedor web Apache Tomcat

para el despliegue de aplicaciones web. Incluye herramientas para el desarrollo en lenguaje Java, Groovy, y otros, prevé de posibles cambios o migración de lenguaje o sistema operativo, los cuales no afectaría al equipo de desarrollo (22).

TortoiseSVN 1.8.8

TortoiseSVN es un cliente de código abierto, gratuita para el sistema de control de versiones de Apache Subversion[3]. Los archivos se almacenan a lo largo del tiempo en un repositorio central, el cual es como un servidor de archivos ordinario, salvo que recuerda todos los cambios hechos a sus archivos y directorios. Esto le permite recuperar versiones antiguas de sus ficheros y examinar la historia de cómo y cuándo cambiaron sus datos, y quién lo cambió. Subversion, es un sistema general que puede ser utilizado para manejar cualquier colección de archivos, incluyendo el código fuente. Promueve la colaboración del equipo de trabajo, puesto que diversos usuarios desde donde estén pueden modificar y administrar el mismo conjunto de datos; gracias a que la herramienta puede crear repositorios de ficheros, los cuales pueden ser accedidos mediante la red desde distintos ordenadores (23).

Sistema Gestor de Base de Datos

Un Sistema Gestor de Bases de Datos (SGBD) es un conjunto de programas que acceden y gestionan una colección de datos relacionados entre sí, estructurados y organizados, que se denominan Base de Datos; permite definir los datos a distintos niveles de abstracción y manipularlos, garantizando la seguridad e integridad de los mismos (24).

PostgreSQL 9.1

PostgreSQL es un sistema de gestión de bases de datos usado en entornos de software libre. Es compatible con una gran parte del estándar SQL y ofrece múltiples características como: consultas complejas, integridad transaccional, etc. Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. Cuenta con un amplio conjunto de tipos de datos, permitiendo además su extensión mediante tipos y operadores definidos y programados por el usuario, permite definir un nuevo tipo de tabla a partir de otra previamente definida (25).

Servidor de Aplicaciones Web

[3] Subversion (SVN): sistema de control de versiones.

Apache Tomcat 7.0.37

Apache Tomcat es un servidor de aplicaciones web con soporte de servlets y de JavaServer Pages (JSP). Se desarrolla en un entorno abierto, permite construir infraestructuras altamente personalizadas a sus necesidades de servicio, y seguro pues mantiene un historial de que nunca ha habido casos reportados de un fallo de seguridad relacionados. Es un contenedor web escrito en Java por lo que funciona en cualquier sistema operativo que disponga de una máquina virtual Java. Además está integrado al framework Grails, permitiendo un rápido despliegue de las aplicaciones web (26).

1.4.4 Tecnologías

JQuery

JQuery es una biblioteca JavaScript, que implementa una serie de clases que funcionan de forma exacta en todas las plataformas más habituales. Ofrece una infraestructura con la que tendremos mayor facilidad para la creación de aplicaciones complejas del lado del cliente (27). Aporta disímiles ventajas como el ahorro de muchas líneas de código y transparencia en el soporte de la aplicación para los navegadores principales. Es una librería pequeña, pero una de las más potentes integradas a Grails, siendo beneficioso en el uso del lenguaje de programación JavaScript, permitiendo mayor facilidad en el desarrollo del sistema. Es un producto documentado (28).

Java Database Connectivity

Java DataBase Connectivity (JDBC), es una Interfaz de programación de aplicaciones (API, por sus siglas en inglés) que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede (29). Esta tecnología es usada para lograr la conexión con la Base de Datos del Gespro, pues este sistema no brinda un servicio web que devuelva los datos de los proyectos terminados, por lo que dicha conexión nos permite el acceso y consulta a los datos.

Marco de Trabajo

Grails 2.2.1

Grails es un framework[4] para el desarrollo de aplicaciones Web, que se basa en el lenguaje dinámico Groovy. Requiere menos código para crear aplicaciones, lo que significa menos errores y menos líneas de código de mantener. Pretende ser un marco de trabajo altamente productivo siguiendo paradigmas tales como:

- **Convención mejor que configuración (CoC):** Aunque en una aplicación Grails existen archivos de configuración, es muy poco probable que tengas que editarlos manualmente ya que el sistema se basa en convenciones (18).
- **DRY: Don't repeat yourself! (¡No te repitas!):** La participación de Spring Container en Grails permite inyección de dependencias mediante IoC (Inversion of Control), de forma que cada actor en la aplicación debe definirse una única vez, haciéndose visible a todos los demás de forma automática (18).

Dichos paradigmas proporcionan un entorno de desarrollo estandarizado y ocultan gran parte de los detalles de configuración al programador. Grails intenta ofrecer el equilibrio adecuado entre consistencia y funcionalidades potentes (18).

Bootstrap 2.2.2

Es un framework de presentación de software libre para el diseño de sitios y aplicaciones web, con numerosos componentes que minimizan el tiempo y el esfuerzo en el desarrollo de dichas aplicaciones. Contiene plantillas de diseño con tipografía, menús de navegación, formularios y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales. Es compatible con la mayoría de los navegadores web (30).

1.5 Conclusiones parciales

En este capítulo se realizó un estudio de conceptos y definiciones asociados al proceso archivístico en la DGP para lograr una mejor comprensión de la problemática planteada. Al mismo tiempo se analizaron un conjunto de sistemas informáticos con el objetivo de realizar un estudio de las funcionalidades comunes a las requeridas para esta investigación. Se describió la colección de tecnologías y herramientas definidas para el desarrollo en el Xabal Arkheia.

[4] Marco de Trabajo. [framework o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos]

Capítulo 2. Características del sistema

2.1 Introducción

En el presente capítulo se ofrece una descripción general de la propuesta de solución para la personalización del Xabal Arkheia para la DGP. Se define también el modelo de negocio donde se describen los procesos del mismo, se especifican los requisitos funcionales y no funcionales del sistema y se ofrece una definición detallada de los casos de uso a partir de su descripción y representación gráfica, y de la definición de los actores.

2.2 Propuesta de solución

Luego de realizar un estudio de la situación problemática y el estado del arte, se propone como solución la personalización del Xabal Arkheia para la DGP, haciendo uso de funcionalidades que conforman los módulos de la versión actual del sistema Arkheia. Con esta propuesta de solución se logrará salvaguardar los productos software creados en la universidad; con toda la información referente al desarrollo de los mismos: la documentación empleada como bibliografía, código fuente del software y la gestión de los proyectos a través de las tareas y usuarios asociados al mismo. Dicha propuesta constituye una nueva forma de organización y control de las producciones en la UCI.

Se propone reutilizar algunas de las funcionalidades ya creadas del módulo Administración:

- Actualizar Menú
- Registrar usuario
- Mostrar usuario
- Mostrar usuario conectado
- Gestionar Nomencladores
- Actualizar configuración
- Gestionar roles
- Rendimiento

Para este módulo se implementan nuevas funcionalidades que responden a requerimientos del cliente:

- *Estado del servidor* que ofrece una panorámica del sistema mostrando datos tales como: descripción, arquitectura y versión del Sistema Operativo, información de la memoria y del CPU, información del sistema de archivos e información de la Máquina Virtual de Java.

- *Configuración de la cuenta de correo electrónico,* el objetivo fundamental de esta funcionalidad es la notificación de los eventos que ocurren en el sistema.
- *Configuración de parámetros para acceso remoto a datos a archivar* que posibilita configurar la conexión a la base de datos del Gespro, de proyectos terminados mediante la tecnología JDBC, requiriendo de los datos: servidor, puerto, usuario, alias, dirección de respuesta y contraseña.

En el módulo Procesamiento también se reutilizarán una serie de funcionalidades:

- Explorar
- Búsqueda general y Búsqueda avanzada(estas dos funcionalidades estaban implementadas en un módulo Búsqueda, ahora pertenecen al módulo Procesamiento)

Y se implementan como nuevas:

- *Describir*, que describe los metadatos según los requisitos de la DGP y posibilita subir uno o más ficheros luego de completar la descripción.
- *Proyectos a archivar,* muestra un listado de los proyectos terminados en la universidad listos para archivar con la opción de Describir entre sus acciones.
- *Proyectos descritos,* permite la búsqueda de proyectos descritos en el sistema.
- *Autores,* permite gestionar los autores y una vez registrados, permite ejecutar las acciones de modificar, eliminar y ver detalles del mismo.

Una peculiaridad de este sistema es que transcurrido dos años luego de registrar el proyecto, debe lanzar una notificación del tiempo que lleva registrado el software, dicha funcionalidad constituye un requerimiento del cliente.

Se propone implementar la autenticación por LDAP o Lightweight Directory Access Protocol (en español Protocolo Ligero de Acceso a Directorios) el cual es un protocolo de acceso unificado a un conjunto de información sobre una red. Habitualmente LDAP, almacena la información de autenticación (usuario y contraseña) y es utilizado para autenticarse aunque es posible almacenar otra información como datos de contacto del usuario, ubicación de diversos recursos de la red, permisos, etc.

En el módulo de clasificación se gestionará el cuadro de clasificación, donde el usuario de la DGP podrá definir toda la estructura lógica de la dirección de proyectos de la UCI. Esta funcionalidad será modificada con respecto a la estructura lógica de Arkheia, cuyo cuadro de clasificación es lineal, lo que significa que

cada tipo de nodo solo pertenece a un tipo de nodo determinado; sin embargo la DGP requiere que la estructura del cuadro de clasificación sea no lineal, para que así cualquier nodo creado no deba pertenecer solo al nodo de un nivel determinado, sino que pueda pertenecer a un nodo de cualquier nivel. Por ejemplo, para este caso en particular, que los centros productivos puedan pertenecer tanto a una Facultad como a la Universidad, como es el caso del Centro de Informática Médica, que pertenece a la DGP.

La gestión de la estructura de ubicación física de la institución, se utilizará como la primera versión de Arkheia, en la que la creación de las unidades de conservación partía desde la gestión de la estructura física y era responsabilidad del coordinador de procesos técnicos, encargado de la ubicación física en la institución archivística. Este proceso no se encuentra definido de esta forma en las versiones actuales del Arkheia, donde es el descriptor de documentos quien crea las unidades de conservación.

Una parte esencial del sistema es el registro de proyectos terminados, en este caso se proponen dos formas: el registro manual y el registro semiautomático de metadatos. El registro manual funciona a través de un formulario que recoge los datos de un proyecto, eliminando el formulario de registro de documentos que usa la versión actual de Arkheia con la norma ISAD (G). Por otra parte para el registro semiautomático, se realizan consultas a datos que están almacenados en otros sistemas, en los que se apoya actualmente la universidad para controlar el desarrollo del software, y en este caso va dirigida en dos direcciones, para lograr este registro se realizará la integración con el eXcriba y el Gespro, explicado con detalles más adelante.

Luego de esta integración, faltaría otro proceso que es la asociación de ficheros a la descripción del sistema, donde por el registro manual se deben introducir todos los datos a través de formularios como se explicaba anteriormente, y por el registro semiautomático ya estaría incluida la documentación solo faltaría el código fuente del proyecto.

La gestión del cuadro de clasificación y la estructura de ubicación física, se pueden realizar de forma simultánea, no importa el orden; sin embargo para describir un documento es imprescindible que se haya creado al menos un cuadro de clasificación y una unidad de conservación.

Integración con el eXcriba y el Gespro

Capítulo 2. Características del sistema

La UCI está llevando a cabo la tarea de redefinir sus procesos de producción de software, con el objetivo de lograr una certificación de nivel 3 del modelo CMMI. Para ello ha orientado las siguientes políticas sobre las herramientas a utilizar en los proyectos:

- La documentación generada en los expedientes de proyecto se gestionan desde el eXcriba.
- Las tareas asignadas a los recursos humanos de los proyectos se gestionan mediante el Gespro, desde donde se puede controlar el desarrollo de los mismos.
- El código fuente de las aplicaciones se versionan con Subversion en un servidor central.

En la actualidad la totalidad de los proyectos cumplen con estas políticas. Desde este punto de vista se realiza la siguiente propuesta de integración en dos pasos del Arkheia con el eXcriba y el Gespro:
En el momento en que concluye un proyecto, se actualiza el estado del mismo en el Gespro, generándose un conjunto de sucesos en aquel sistema enviando los metadatos de un proyecto a su base de datos de proyectos terminados

Paso 1. El sistema Arkheia debe conocer cuando un proyecto se cierra. Para ello se propone la implementación de comunicación entre sistemas mediante servicios web. Esta comunicación puede realizarse de dos formas:

1. Desde el Gespro hacia el Arkheia: Cuando se cierra un proyecto, el Gespro comunica al Arkheia, enviando todos los metadatos necesarios para su registro en el archivo histórico de la UCI.
2. Desde el Arkheia hacia el Gespro: Diariamente el Arkheia consulta por los nuevos proyectos terminados. Al obtener una respuesta positiva, le pide los metadatos necesarios para registrar el proyecto en el archivo histórico de la UCI.

El siguiente paso que conlleva el cierre de un proyecto es el cierre de la documentación asociada a ese proyecto en el eXcriba. No existe relación entre un paso y otro; se puede cerrar la documentación primero y luego el proyecto en el Gespro o viceversa.

Paso 2. Una vez registrado de forma automática los metadatos del proyecto terminado, el sistema consulta en el eXcriba por la documentación asociada mediante el consumo de servicios web. Esta documentación es recibida en un comprimido con extensión .zip.

Al término de este proceso, la integración con estos dos sistemas habrá concluido. Sin embargo esta integración no es posible en la forma ideada por los siguientes factores:

1. No existe un identificador único para un proyecto en los dos sistemas.
2. El Gespro no tiene implementado algún servicio web que brinde la información de los proyectos terminados, ni personal para implementarlo.

Esta situación conllevó a buscar una nueva solución para poder extraer los metadatos de los proyectos automáticamente. La dirección de la DGP realiza consultas históricas sobre una base de datos de proyectos terminados. Esta base de datos se nutre directamente del Gespro. De esta forma se puede llevar a cabo el paso 1 de la integración con la variante 2, ahora realizando consultas periódicas a esta base de datos.

2.3 Modelo de negocio

El modelado del negocio es una técnica para comprender los procesos del negocio y su objetivo es identificar los casos de uso del software y las entidades de negocio relevantes que el software debe soportar, de forma que se podría modelar sólo lo necesario para comprender el contexto. Describe los procesos de negocio en términos de casos de uso y actores del negocio que se corresponden con los procesos del negocio y los clientes respectivamente (13).

La siguiente imagen describe el proceso de negocio del módulo Procesamiento, que inicia con la descripción de documentos. Para describir un software se realizan primeramente dos tareas en paralelo que son: asignar la ubicación física gestionando la unidad de conservación; y la gestión del cuadro de clasificación que es la ubicación lógica, donde se van a poder ubicar y clasificar los documentos. Luego de la ejecución de estas tareas simultáneas se procede a asignar los ficheros, en caso de que exista algún archivo se asigna y se guarda su descripción y termina el proceso; en caso contrario que no existan archivos para asignar se guarda la descripción y culmina el proceso.

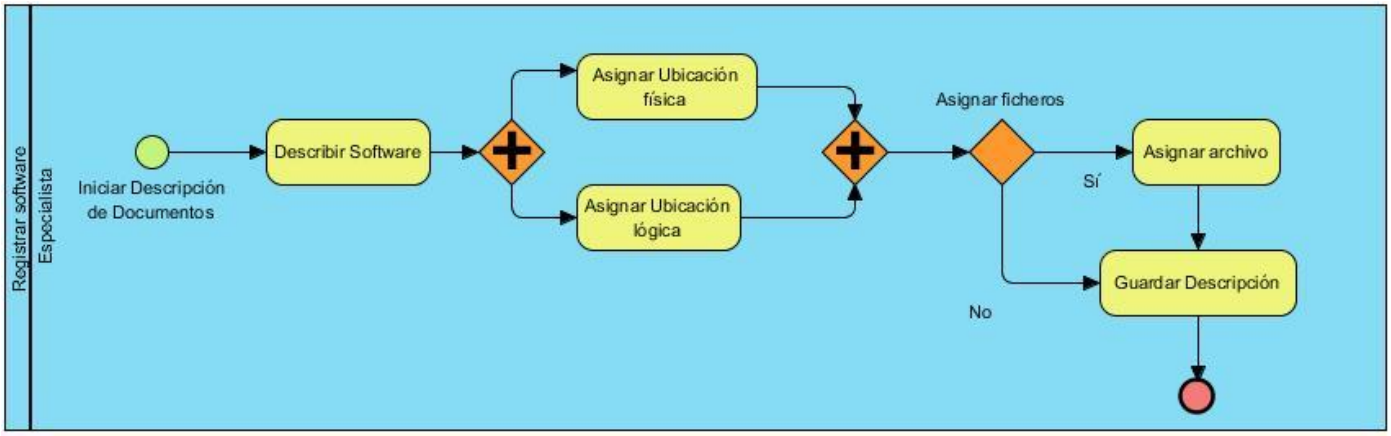


Figura 1. Diagrama de proceso de módulo Procesamiento

2.4 Especificación de los requisitos de software

Los requisitos de software son las condiciones o capacidades que el sistema debe cumplir para llegar a un acuerdo entre las partes sobre qué debe hacer y cómo debe ser el sistema. El flujo de trabajo de los requisitos basa su propósito fundamental en guiar el desarrollo hacia el sistema correcto, lo que se consigue mediante una descripción suficientemente buena de los mismos. Están direccionados tanto al cliente como al desarrollador, y el lenguaje utilizado para su redacción debe ser sencillo, de forma que sea fácilmente comprensible para todos los involucrados (13).

2.4.1 Requisitos Funcionales

Expresan la naturaleza del funcionamiento del sistema, es decir, cómo interacciona el sistema con su entorno y cuáles van a ser su estado y funcionamiento (13). A continuación se detallan los requisitos funcionales del sistema, de los cuales una gran parte responden a la versión actual de Xabal Arkheia que se va a personalizar para la DGP, para este caso constituyen nuevos requisitos los pertenecientes al módulo Integración y Almacenamiento, así como algunos que forman parte del resto de los módulos.

Módulo Configuración

RF1. Registrar datos de la institución.

RF2. Definir estructura física de la institución.

RF3. Crear cuenta de administración.

RF4. Establecer parámetros de configuración del sistema.

Módulo Administración

RF5. Autenticar un usuario para autorizar su acceso al sistema.

RF6. Cerrar la sesión de un usuario iniciada en el sistema.

RF7. Cargar la configuración del usuario en correspondencia con sus privilegios.

RF8. Buscar los usuarios registrados a partir de los criterios de búsquedas introducidos.

RF9. Registrar los datos de un usuario.

RF10. Modificar los datos de un usuario.

RF11. Activar los datos de un usuario.

RF12. Desactivar los datos de un usuario.

RF13. Visualizar los datos de detalle de un usuario.

RF14. Eliminar los datos de un usuario.

RF15. Registrar los datos de un nomenclador.

RF16. Modificar los datos de un nomenclador.

RF17. Eliminar los datos de un nomenclador.

RF18. Establecer la contraseña de un usuario.

RF19. Cambiar la contraseña de un usuario.

RF20. Registrar los datos de un rol.

RF21. Modificar los datos de un rol.

RF22. Eliminar los datos de un rol.

RF23. Mostrar los usuarios conectados al sistema.

RF24. Liberar memoria del contenedor de aplicaciones.

RF25. Actualizar la configuración de los parámetros generales necesarios para el correcto funcionamiento del sistema.

RF26. Actualizar posiciones de las opciones del menú.

Módulo Clasificación

RF27. Explorar el cuadro de clasificación.

RF28. Adicionar los datos de un nivel del cuadro de clasificación.

RF29. Actualizar los datos de un nivel del cuadro de clasificación.

RF30. Eliminar un nivel del cuadro de clasificación.

RF31. Actualizar la descripción de un nivel del cuadro de clasificación.

RF32. Visualizar la ubicación lógica de un elemento del cuadro de clasificación.

RF33. Cambiar la ubicación lógica de un elemento del cuadro de clasificación.

RF34. Crear cuadro de clasificación de un tipo de cuadro.

Módulo Procesamiento

RF35. Describir un proyecto.

RF36. Seleccionar la ubicación lógica de un documento.

RF37. Asignar ficheros a una descripción de documento.

RF38. Eliminar ficheros de la descripción.

RF39. Modificar los datos de la descripción de un proyecto.

RF40. Visualizar los datos de detalles de la descripción de un proyecto

RF41. Buscar las descripciones registradas a partir de los criterios de búsquedas introducidos.

RF42. Registrar los datos de un autor.

RF43. Modificar los datos de un autor.

RF44. Mostrar datos de un autor.

RF45. Mostrar documentos descritos.

Módulo Almacenamiento

RF46. Adicionar los datos de un nodo de la estructura de ubicación física.

RF47. Modificar los datos de un nodo de la estructura de ubicación física.

RF48. Mover un nodo de la estructura de ubicación física.

RF49. Visualizar los datos de detalle de un nodo de la estructura de ubicación física.

RF50. Eliminar un nodo de la estructura de ubicación física.

RF51. Explorar la estructura de ubicación física.

RF52. Adicionar una categoría a la estructura de ubicación física.

RF53. Modificar una categoría de la estructura de ubicación física.

RF54. Eliminar una categoría de la estructura de ubicación física.

RF55. Registrar los datos de una unidad de conservación de la estructura de ubicación física.

RF56. Modificar los datos de una unidad de conservación de la estructura de ubicación física.

RF57. Cambiar la ubicación física de una unidad de conservación de la estructura de ubicación física.

RF58. Cambiar el estado de una unidad de conservación.

RF59. Visualizar los datos de detalle de una unidad de conservación de la estructura de ubicación física.
RF60. Eliminar una unidad de conservación de la estructura de ubicación física.
RF61. Configurar estructura de ubicación física.

Módulo Integración

RF62. Actualizar los nomencladores desde el GESPRO.
RF63. Cargar datos de los proyectos archivados en el Gespro según entidad desarrolladora.
RF64. Emitir alerta una vez cumplido dos años a partir de la fecha de ser archivado.
RF65. Proponer ubicación física para proyectos que deben ser archivados.
RF66. Crear un panel de proyectos para archivar.
RF67. Emitir una alerta cuando se adicione un proyecto al panel.
RF68. Archivar la documentación del proyecto gestionada en el eXcriba.
RF69. Eliminar documentación del proyecto en el eXcriba.
RF70. Mostrar en un panel los recursos del servidor.
RF71. Emitir alerta sobre los recursos del servidor.
RF72. Implementar autenticación por DNS.
RF73. Modificar descripción de proyecto.
RF74. Modificar ficheros del proyecto.

2.4.2 Requisitos no Funcionales

Los requisitos no funcionales especifican propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma, facilidad de mantenimiento, extensibilidad y fiabilidad (13).

Requerimientos de Hardware

RnF1. PC Cliente

- RAM: 512 MB
- Tarjeta de Red

RnF2. PC Servidor

- RAM: 1 GB para servidor de Base de Datos.

- RAM: 2 GB para servidor de Aplicaciones.
- Disco Duro: 40 GB para Base de Datos y 1 TB para Aplicaciones.
- Tarjeta de Red

Para determinar los requerimientos de hardware se realizó un estudio para un escenario real tomando en cuenta cómo se desarrolla el trabajo en la DGP y cada que margen de tiempo se registra un proyecto terminado en la UCI, lo que queda fundamentado en los siguientes resultados. Ver Anexo 1

Requerimientos de Software

RnF3. PC Cliente

- Navegador Web: Mozilla Firefox, Google Chrome, Opera, Safari.
- Lector de pdf.

RnF4. PC Servidor

- Servidor de Aplicaciones Web: Apache Tomcat 7.0.37.
- Servidor de Base de Datos: Postgresql 9.1.
- Sistema Operativo: Windows, Linux.

Requerimientos de apariencia o interfaz externa

RnF5. El sistema sigue las pautas de diseño establecidos para la línea de productos XABAL referente a sistemas de administración pública realizados en la UCI.

Restricciones de diseño e implementación

RnF6.

- Framework de desarrollo Grails 2.2.1.
- Framework de Presentación Bootstrap 2.2.2.
- Arquitectura Modelo Vista Controlador.

Requerimientos de Seguridad

RnF7. La información controlada por el sistema debe estar protegida de acceso no autorizado para lo cual se establece un nivel de acceso a la aplicación mediante la gestión de roles. Cada grupo tendrá su perfil específico donde se otorgarán privilegios y accesos a las diferentes secciones del sistema. El perfil de los usuarios y administradores será en base a los cargos existentes en la DGP.

Usabilidad

RnF8. El sistema debe ser intuitivo, fácil de aprender por los usuarios y su manejo debe ser eficiente. A través de formularios y listados, la captura, edición y extracción de la información deberá ser fácil de realizar.

RnF9. El sistema debe tener una interfaz gráfica sencilla. Este requisito ayudará al usuario a poder localizar las herramientas necesarias para desarrollar su trabajo sin necesidad de un extensa capacitación ni de continua asesoría, el uso del sistema deberá ser fácil para el acceso a la información sin necesidad de tener conocimientos técnicos avanzados en tecnologías de la información, únicamente se requerirá el conocimiento básico de Windows e Internet.

Requerimientos de Rendimiento

Eficiencia

RnF10. Tiempo de respuesta por transacción (promedio, máximo). El sistema debe tener un tiempo de respuesta promedio de 850ms por transacción y un máximo de 1500ms.

RnF11. Rendimiento. El sistema debe permitir 5 transacciones por segundo como mínimo.

RnF12. Capacidad. El sistema debe permitir más de 20 usuarios conectados.

RnF13. Utilización de recursos. El promedio de utilización de memoria debe ser de 650 MB. Disco duro del servidor: 1 TB o superior para almacenar el sistema de archivos de la aplicación. Ver Anexo 2.

Soporte

Se llevan a cabo las acciones a tomar una vez que se ha terminado el desarrollo del sistema con motivo de lograr su mejoramiento progresivo y evolución en el tiempo para que el cliente se sienta satisfecho. El soporte del sistema incluye:

RnF14. Pruebas Pilotos. Se realizan para probar el software en el ambiente real en un determinado tiempo.

RnF15. Peticiones de cambio. Cuando se realizan las pruebas pilotos pueden existir las peticiones de cambio, lo cual es una solicitud del cliente para la modificación de alguna funcionalidad del sistema.

RnF16. Mantenimiento. Es la solución de problemas que se presenten en el software por errores de la implementación del código.

RnF17. Instalación y Configuración. El equipo de desarrollo se encarga de la instalación y configuración del servidor web y servidor de base de datos.

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

RnF18. Entregar manuales de usuario. Los manuales de usuario describen las acciones de cada módulo del sistema.

Requerimientos de Portabilidad

RnF19. El sistema podrá ser usado bajo cualquier sistema operativo ya sea Linux o Windows.

2.5 Definición de los Casos de Uso

Los casos de uso son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores, es decir, especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores (13).

2.5.1 Definición de los actores

Un Actor representa un rol que los usuarios juegan al interactuar con el sistema, ellos son siempre externos al sistema que se modela. Representan el entorno del sistema. El rol puede ser desempeñado por personas, dispositivos (hardware) u otros sistemas, y una misma persona, dispositivo, sistema puede desempeñar varios roles. Los actores del sistema son los encargados de automatizar las actividades del sistema e interactuar con el mismo (31). A continuación se muestran los actores que intervienen en el sistema y una breve descripción de las tareas que realizan en cada módulo.

Actores	Descripción
Administrador	• Configuración: instala y configura el sistema, configura los parámetros generales y crea la cuenta de administración. • Administración: gestiona usuario, nomenclador, contraseña, roles, busca y muestra usuarios conectados o no, actualiza configuración, libera memoria en el servidor de la aplicación y actualiza la posición de las opciones del menú del sistema. • Almacenamiento: configura estructura de Ubicación Física.
Especialista	• Procesamiento: describe proyecto, gestiona ficheros asociados al proyecto, gestiona actores, modifica y busca descripción.

Usuario	• Procesamiento: ve detalles de Descripción de Proyecto. • Administración: ve detalles del usuario autenticado en el sistema, cambia la contraseña, se autentica en el sistema y cierra la sesión.
Supervisor	• Clasificación: gestiona niveles de cuadro de clasificación, explora cuadro de clasificación, gestiona ubicación lógica dentro de un cuadro de clasificación y registra cuadros de clasificación.
Coordinador de almacenamiento	• Almacenamiento: gestiona, explora y edita configuración de estructura de ubicación física y gestiona unidad de conservación.

2.5.2 Diagrama de casos de uso

El diagrama de casos de uso representa la forma en como un Cliente (Actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan (31).

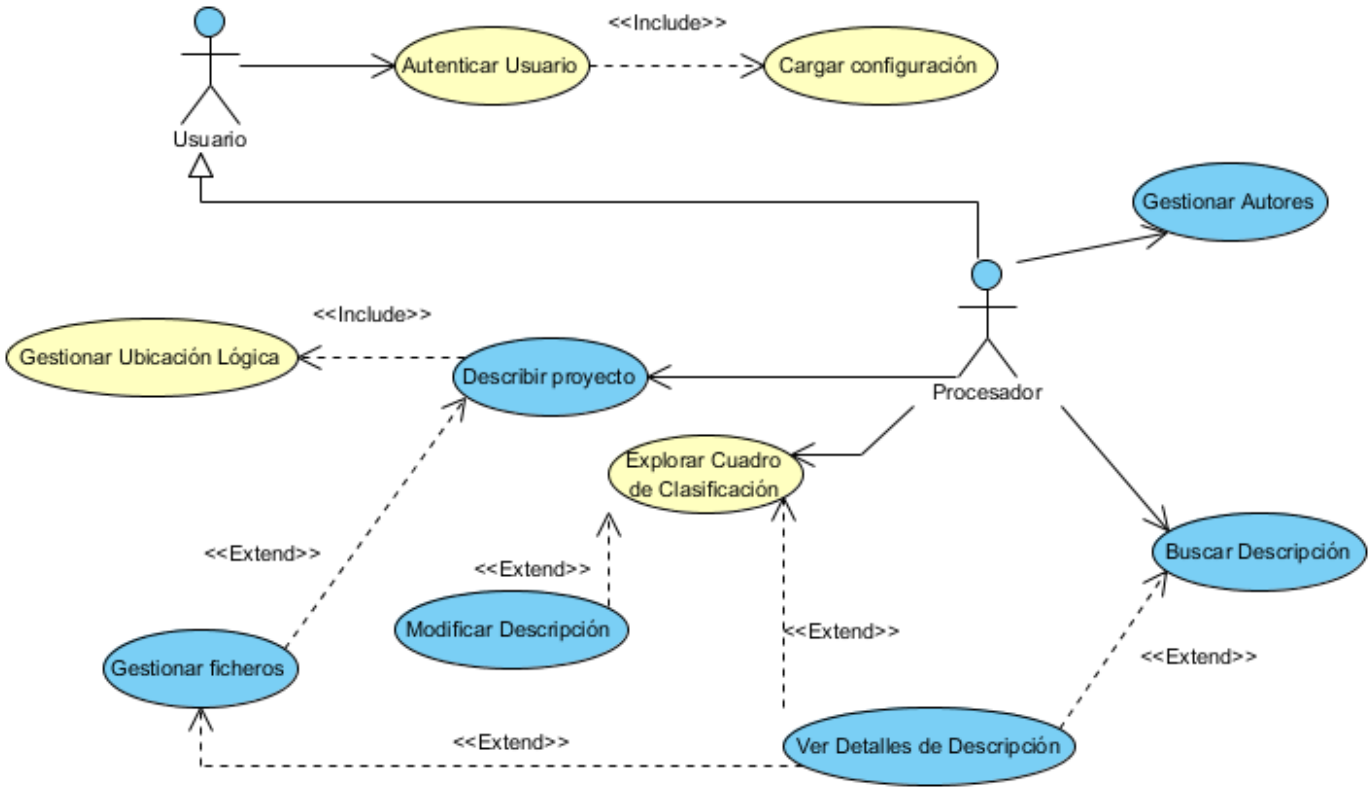


Figura 2. Diagrama de casos de uso del módulo Procesamiento

2.5.3 Descripción de los Casos de Uso

La descripción de los casos de uso proporciona una detallada descripción que se documenta con texto informal, la misma ofrece una lista de los pasos que sigue el actor para interactuar con el sistema (31).
A continuación se describen brevemente los casos de uso del módulo Procesamiento:

CU 1. Describir Proyecto.

Objetivo	Describir proyecto.	
Actores	Especialista.	
Resumen	El Especialista solicita realizar una acción sobre la descripción de un proyecto, el sistema muestra el formulario que permite realizar la acción solicitada. La acción es: describir. El Especialista indica las operaciones a realizar por el sistema resultando la descripción del proyecto almacenada en la BD.	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	El Especialista está autenticado. Existe un cuadro de clasificación creado. Existe una unidad de conservación.	
Postcondiciones	Datos de la descripción registrados en la BD.	
Flujo de eventos		
Flujo básico Describir Proyecto		
	Actor	**Sistema**
1.	Selecciona la opción del menú principal “Describir”.	
2.		Muestra un formulario solicitando los siguientes datos: • Nombre oficial.(obligatorio) • Número de registro. • Nombre corto o

		identificador.(obligatorio) • Cliente.(obligatorio) • Autores.(obligatorio) • Objetivo general.(obligatorio) • Alcance.(obligatorio) • Entidad desarrolladora.(obligatorio) • Fecha inicial.(obligatorio) • Fecha final.(obligatorio) • Tipo de artefacto.(obligatorio) • Clasificación del cliente. • Naturaleza. • Impacto.(obligatorio) • Estado de completitud.(obligatorio) • Ubicación Lógica. (obligatorio). **Ver CU Gestionar Ubicación Lógica.** • Resumen técnico. • Comentarios o valoraciones. • Unidad de conservación. (obligatorio) Muestra las siguientes opciones: • Aceptar. • Aceptar y asociar ficheros • Cancelar
3.	Introduce los datos en el formulario.	
4.	Selecciona la opción “Aceptar”.	
5.		Comprueba que se han introducido los campos obligatorios.
6.		Comprueba que los datos introducidos son correctos. **Ver Modelo Conceptual.**

7.		Almacena en la BD los datos introducidos.
8.		Muestra un mensaje indicando que se han guardado los datos satisfactoriamente.
9.		Termina el caso de uso.
Flujos alternos		
Nº 4a El actor selecciona la opción "Aceptar y asociar ficheros"		
	Actor	**Sistema**
1.	Selecciona la opción "Aceptar y asociar ficheros"	
2.		Comprueba que se han introducido los campos obligatorios.
3.		Comprueba que los datos introducidos son correctos. **Ver Modelo Conceptual.**
4.		Almacena en la BD los datos introducidos.
5.		Muestra un mensaje indicando que se han guardado los datos satisfactoriamente.
6.		**Se ejecuta el CU Gestionar fichero**
7.		Termina el caso de uso.
Nº 4b El actor selecciona la opción "Cancelar"		
	Actor	**Sistema**
1.	Selecciona la opción "Cancelar"	
		Muestra la vista que le dio origen al CU
		Termina el caso de uso.
Nº 5a Datos Obligatorios Vacíos		
	Actor	**Sistema**
1.		Comprueba que se han introducido los datos obligatorios.

2.		Muestra un mensaje indicando que faltan datos por introducir y señala los datos que faltan por introducir. Ir a la acción 3 del flujo básico.
Nº 6a Datos Introducidos Incorrectos		
	Actor	**Sistema**
1.		Comprueba que los datos introducidos son incorrectos (Ver Modelo Conceptual).
2.		Muestra un mensaje indicando que existen datos incorrectos, selecciona los datos que están incorrectos. Los datos se mantienen para que el usuario los modifique si lo desea. Ir a la acción 3 del flujo básico.

Tabla 1 Descripción de CU Describir Proyecto

CU 2. Gestionar ficheros.

Objetivo	Gestionar de una descripción los ficheros asociados al proyecto.(Adjuntar, eliminar, visualizar)
Actores	Especialista
Resumen	Permite al Especialista adjuntar o eliminar de una descripción los ficheros asociadas al proyecto.

Tabla 2 Descripción de CU Gestionar ficheros

CU 3. Ver Detalles de Descripción de Proyecto

Objetivo	Ver los detalles de una descripción de proyecto en el sistema.
Actores	Usuario: (Inicia) Ver los detalles de una descripción de proyecto en el sistema.
Resumen	El CU inicia cuando el Actor selecciona la opción que permite ver los detalles de una descripción de proyecto determinada en el sistema. Se muestra un formulario con los elementos obtenidos y la opción que permite regresar a la página anterior. Termina el caso de uso.

Tabla 3 Descripción de CU Ver detalles de Descripción de proyecto

CU 4. Gestionar Autores

Objetivo	Registrar, modificar y eliminar los autores en el sistema.
Actores	Especialista: (Inicia) Registra, modifica y eliminar los autores.
Resumen	El caso de uso inicia cuando el Especialista decide realizar las acciones permitidas sobre un autor. Las acciones son: registrar, modificar y eliminar un autor. El sistema permite realizar la acción solicitada por el Especialista y termina el caso de uso.

Tabla 4 Descripción de CU Gestionar autores

CU 5. Modificar Descripción

Objetivo	Modificar los datos de la descripción de un proyecto.
Actores	Procesador.
Resumen	El procesador solicita modificar la descripción de un proyecto, el sistema muestra el formulario que permite realizar la acción solicitada. El procesador modifica la descripción del proyecto y guarda los datos en la base de datos.

Tabla 5 Descripción de CU Modificar Descripción

CU 6. Buscar Descripción

Objetivo	Buscar los datos de la descripción de un proyecto.
Actores	Procesador.
Resumen	El caso de uso inicia cuando el procesador decide realizar la búsqueda de la descripción de un proyecto, el sistema muestra el formulario que permite realizar la acción solicitada. La búsqueda puede ser general o avanzada.

Tabla 6 Descripción de CU Buscar Descripción

Modelo conceptual

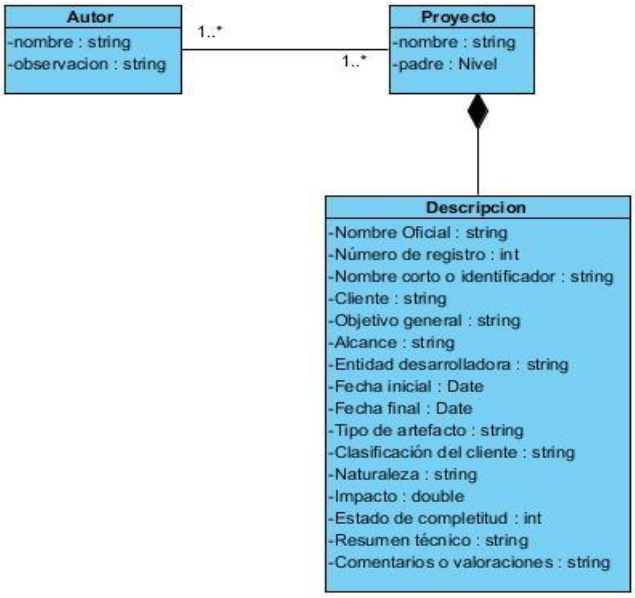


Figura 3 Modelo conceptual de CU Describir proyecto

Prototipo de Interfaz de Usuario

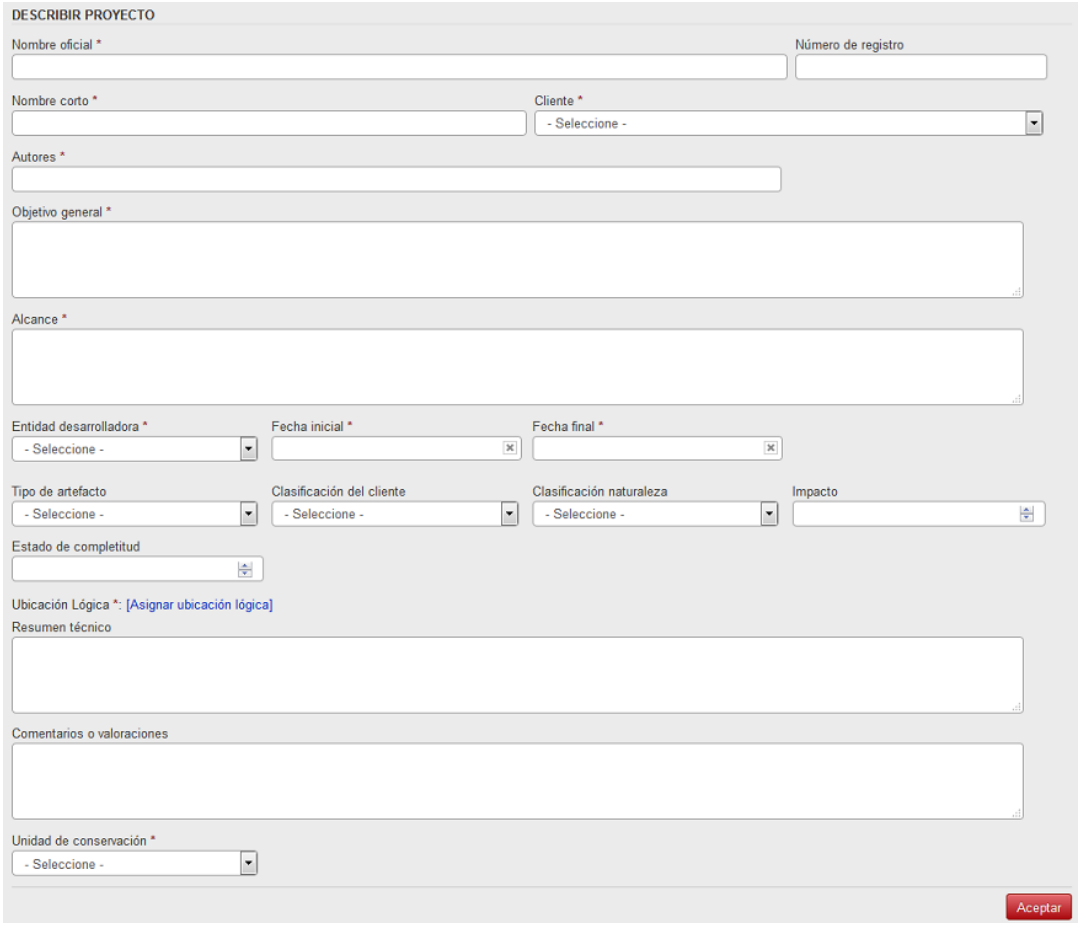


Figura 4 Prototipo de Interfaz de Usuario de CU Describir proyecto

2.6 Conclusiones parciales

El desarrollo del recién concluido capítulo permitió realizar un análisis del problema planteado, y ofrecer una descripción general de la propuesta de solución para la personalización del Xabal Arkheia para la Dirección General de la Producción. Se definió el modelo de negocio donde se describen los procesos del mismo, se especifican los requisitos funcionales y no funcionales del sistema y se ofreció una definición de los casos de uso del módulo Procesamiento a partir de su descripción y representación gráfica. También se definieron los actores que interactúan con el sistema y una breve descripción de las tareas que realizan.

Capítulo 3. Análisis y diseño del sistema

3.1 Introducción

En este capítulo se realiza el análisis del sistema y se desarrolla su diseño a partir de los tipos de diagramas de estructura e interacción, entre los que destacan diagramas de clases, que describen la estructura de un sistema mostrando sus clases, orientados a objetos e 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. En este capítulo se trabaja solamente con un caso de uso del módulo Procesamiento, para hacer menos extensivo y complejo el presente trabajo.

3.2 Análisis

El lenguaje utilizado en el análisis se basa en un modelo de objetos conceptual que se denomina modelo de análisis, este permite el refinamiento y estructuración de los requisitos para lograr una mayor formalización de ellos, también propicia el razonamiento sobre los aspectos internos del sistema y ofrece un mayor poder expresivo. Este modelo brinda una vista interna del sistema y es utilizado fundamentalmente para comprender cómo debería ser diseñado e implementado el mismo. Se considera como una primera aproximación al diseño (13).

3.2.1 Diagramas de clases

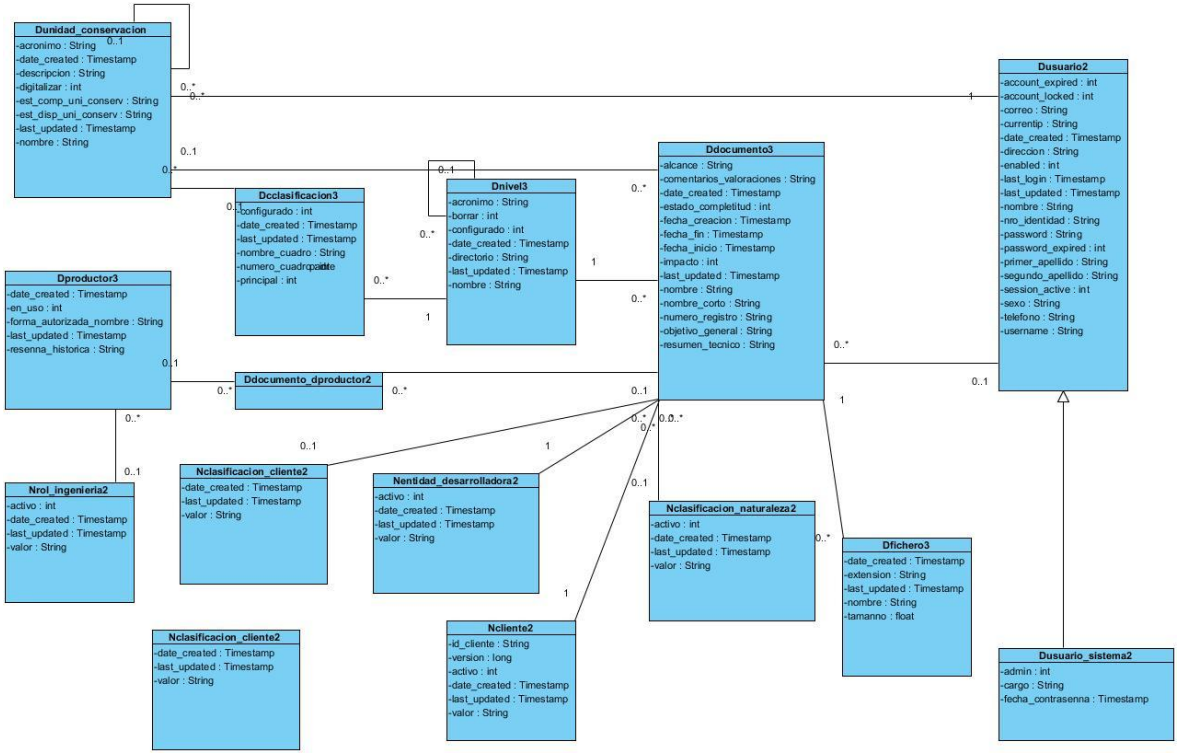


Figura 5 Diagrama de Clases de CU Describir Proyecto del Módulo Procesamiento

3.3 Diseño

En el diseño modelamos el sistema y encontramos su forma para que soporte todos los requisitos, incluyendo los no funcionales y otras restricciones. La entrada en el diseño es el resultado del análisis, pues el modelo de análisis proporciona una comprensión detallada de los requisitos e impone una estructura del sistema que debe tratar de conservarse al máximo cuando se de forma al sistema. El diseño es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción, esto contribuye a una arquitectura estable y sólida y a crear un plano del modelo de implementación (13).

3.3.1 Arquitectura del Sistema

La arquitectura de software de un sistema o programa de computación es la estructura de las estructuras del sistema, la cual comprende los componentes del software, las propiedades de esos componentes

visibles externamente, y las relaciones entre ellos. La arquitectura constituye la representación que capacita al ingeniero del software para:

- analizar la efectividad del diseño para la consecución de los requisitos fijados,
- considerar las alternativas arquitectónicas en una etapa en la cual hacer cambios en el diseño es relativamente fácil, y
- reducir los riesgos asociados a la construcción del software (32).

La arquitectura adoptada para la personalización del Xabal Arkheia para la DGP es la que se establece en la versión actual de Arkheia, la cual se representa como una arquitectura en capas sobre la máquina virtual de Java (JVM), programada con los lenguajes de programación Groovy y Java, en el servidor, montada sobre el framework Grails en la que se definen:

- La capa web: cuyos componentes son los responsables de interactuar y construir la interfaz de usuario.

  En el lado del servidor están presentes:

  - las páginas servidoras de Groovy (GSP) encargadas de construir el HTML,
  - los controladores,
  - la implementación de los flujos web (webflows),
  - el mapeo de las URL del sistema,
  - los filtros sobre las peticiones al sistema, y
  - las librerías de etiquetas.

  En el lado del cliente, el HTML de la aplicación se integra con JQuery, como librería JavaScript (33).
- La capa de servicios: que implementa la lógica del negocio (33).
- La capa de datos: responsable de la gestión y el almacenamiento de la información manejada en el sistema. Encargada de las transacciones a la base de datos y la gestión del sistema de ficheros (33).

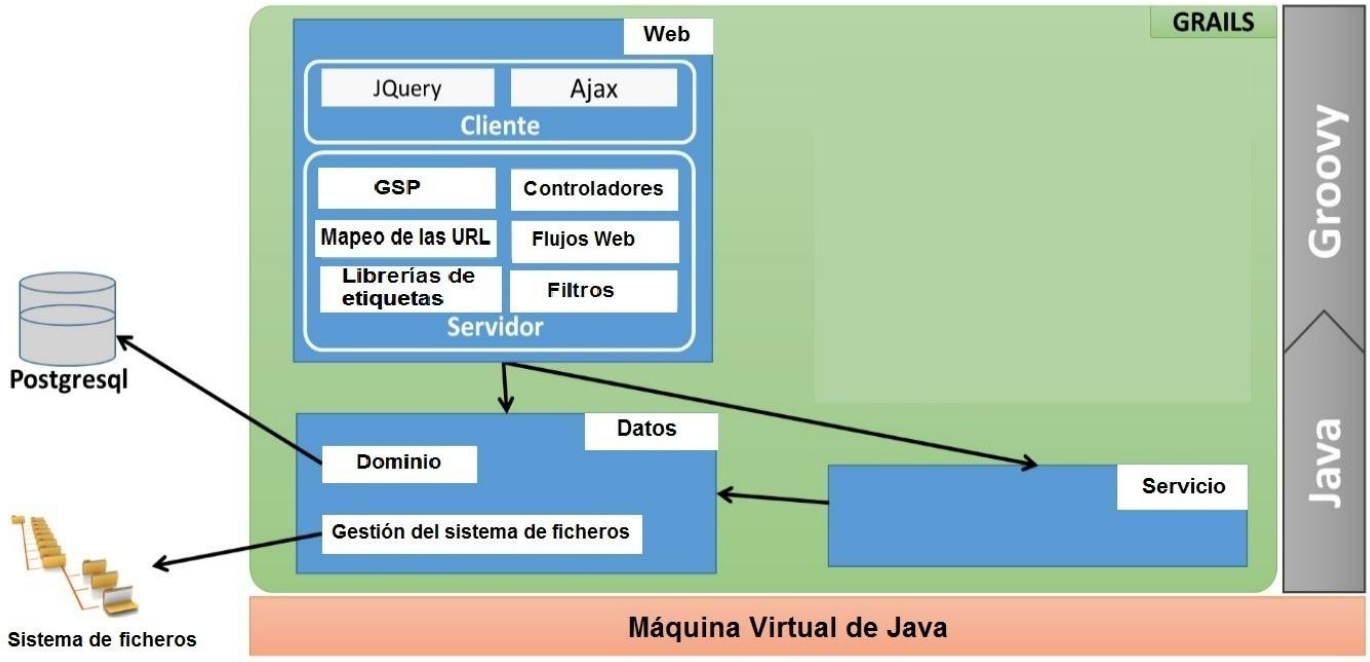


Figura 6. Arquitectura del Sistema

Patrón de Arquitectura

Los patrones arquitectónicos, o patrones de arquitectura, son patrones de software que ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. Resuelven problemas arquitectónicos, adaptabilidad a requerimientos cambiantes, performance, modularidad, acoplamiento, etc. La solución que plantea es la creación de patrones de llamadas entre objetos, decisiones y criterios arquitectónicos, empaquetado de funcionalidad. Este tipo de patrones se utilizan en la fase de desarrollo, en el diseño inicial (34).

Respecto al diseño, Grails sigue el patrón Modelo-Vista-Controlador (MVC), el cual provee un mecanismo para construir una capa web estableciendo que los componentes de una aplicación web deben organizarse en 3 capas distintas según su misión:

- *Modelo*, *o capa de datos*: contiene los componentes que representan y gestionan los datos manejados por la aplicación. En el caso más típico, los objetos encargados de leer y escribir en la base de datos (33).

- *Vista o capa de presentación*: los componentes de esta capa son responsables de mostrar al usuario el estado actual del modelo de datos y presentarle las distintas acciones disponibles (33).
- *Capa de control*: contiene los componentes que reciben las órdenes del usuario, gestionan la aplicación de la lógica de negocio sobre el modelo de datos, y determinan qué vista debe mostrarse a continuación (33).

3.3.2 Patrones utilizados en el diseño

Los diseñadores expertos en orientación a objetos (y también otros diseñadores de software) van formando un amplio repertorio de principios generales y de expresiones que los guían al crear software y reciben el nombre de patrones. En la terminología de objetos, el patrón es una descripción de un problema y su solución que recibe un nombre y que puede emplearse en otros contextos, con una sugerencia sobre la manera de usarlo en situaciones nuevas (14).

Patrones GRASP

GRASP (General Responsibility Asignment Software Patterns), patrones generales de software para asignar responsabilidades que describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Los patrones Experto, Creador, Alta Cohesión, Bajo Acoplamiento y Controlador constituyen patrones GRASP y se refieren a cuestiones y aspectos fundamentales del diseño (14).

- *Experto*: Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.
- *Creador:* Asignarle a la clase B la responsabilidad de crear una instancia de clase A.
- *Alta Cohesión:* Asignar una responsabilidad de modo que la cohesión siga siendo alta.
- *Bajo Acoplamiento:* Asignar una responsabilidad para mantener bajo acoplamiento.
- *Controlador:* Asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase.

3.3.3 Diagramas de clases de Diseño

A continuación se muestra el diagrama de clases del diseño con estereotipos web del caso de uso Describir proyecto del Módulo Procesamiento que detalla el proceso para la descripción de un producto software; que

representa al controlador que ordena a las GSP mostrar los formularios al usuario para que introduzca los datos y luego procesarlos ordenando al servicio guardar documentos y ficheros. El diagrama de clases del diseño del CU Gestionar ficheros se puede ver más adelante. Ver Anexo 3.

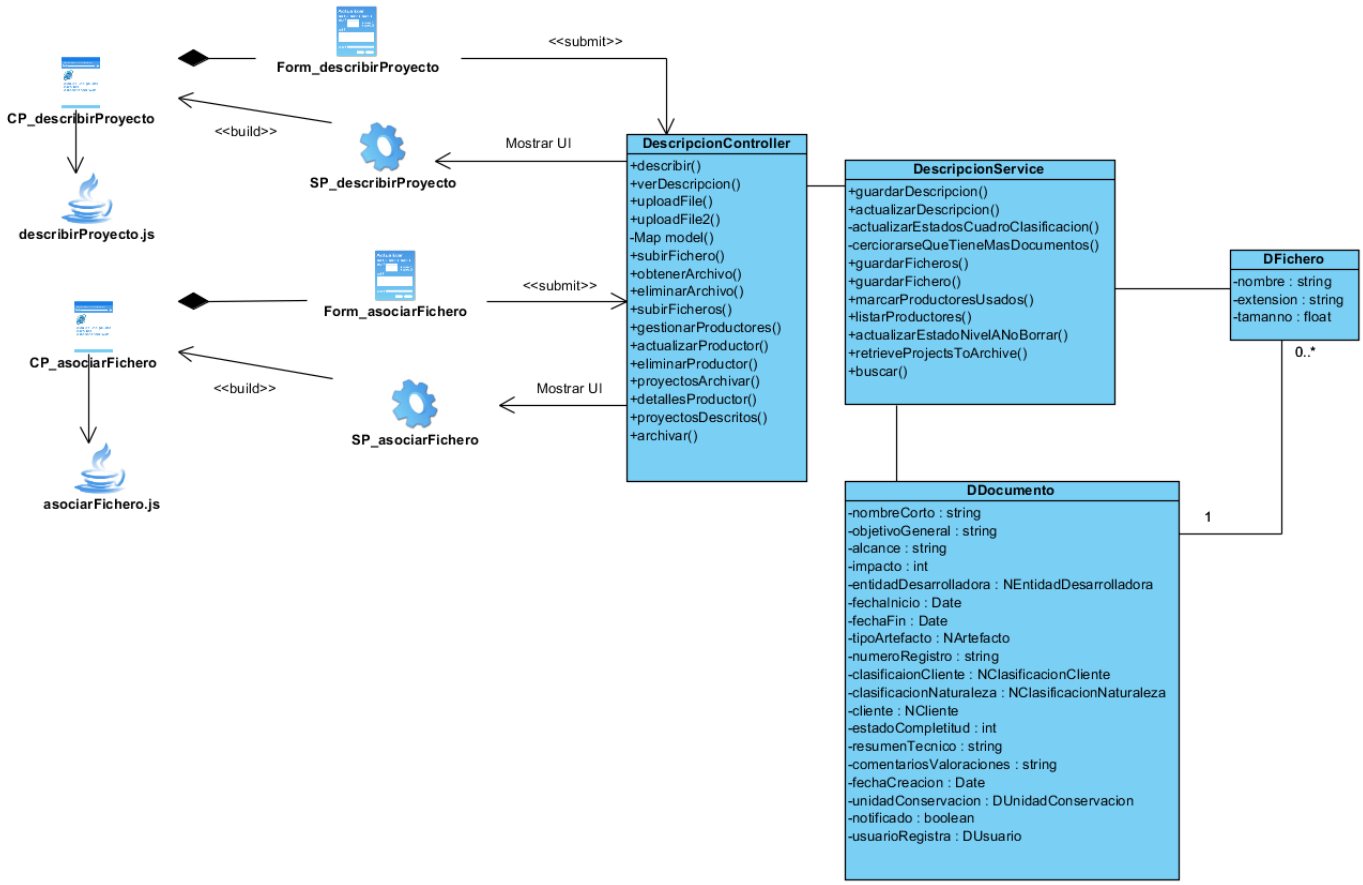


Figura 7 Diagrama de clases del diseño con estereotipos web del CU Describir Proyecto del Módulo Procesamiento

3.3.4 Diagramas de interacción

Los diagramas de interacción describen el comportamiento de un sistema, para demostrar cómo los objetos interactúan dinámicamente en diferentes momentos durante la ejecución del sistema. Los objetos dentro de un sistema se comunican unos con otros, enviándose mensajes, así pues la dinámica de un sistema puede referirse en un caso a cómo dichos objetos colaboran a través de la comunicación que se representa mediante dichos diagramas, que a su vez agrupan a dos tipos: secuencia y colaboración (35).

- Diagrama de secuencia: muestra la interacción de un conjunto de objetos a través del tiempo, es decir, nos proporciona la interacción entre los objetos que se sucede en el tiempo, para un escenario específico durante la ejecución del sistema (35).
- Diagrama de colaboración o comunicación: muestra la interacción de un conjunto de objetos en el espacio, esta descripción se centra en la organización estructural de los objetos que envían y reciben mensajes (35).

Para representar el CU en estudio del módulo Procesamiento se escogió el diagrama de colaboración que implementa los enlaces mediante el paso de mensajes de un objeto a otro.

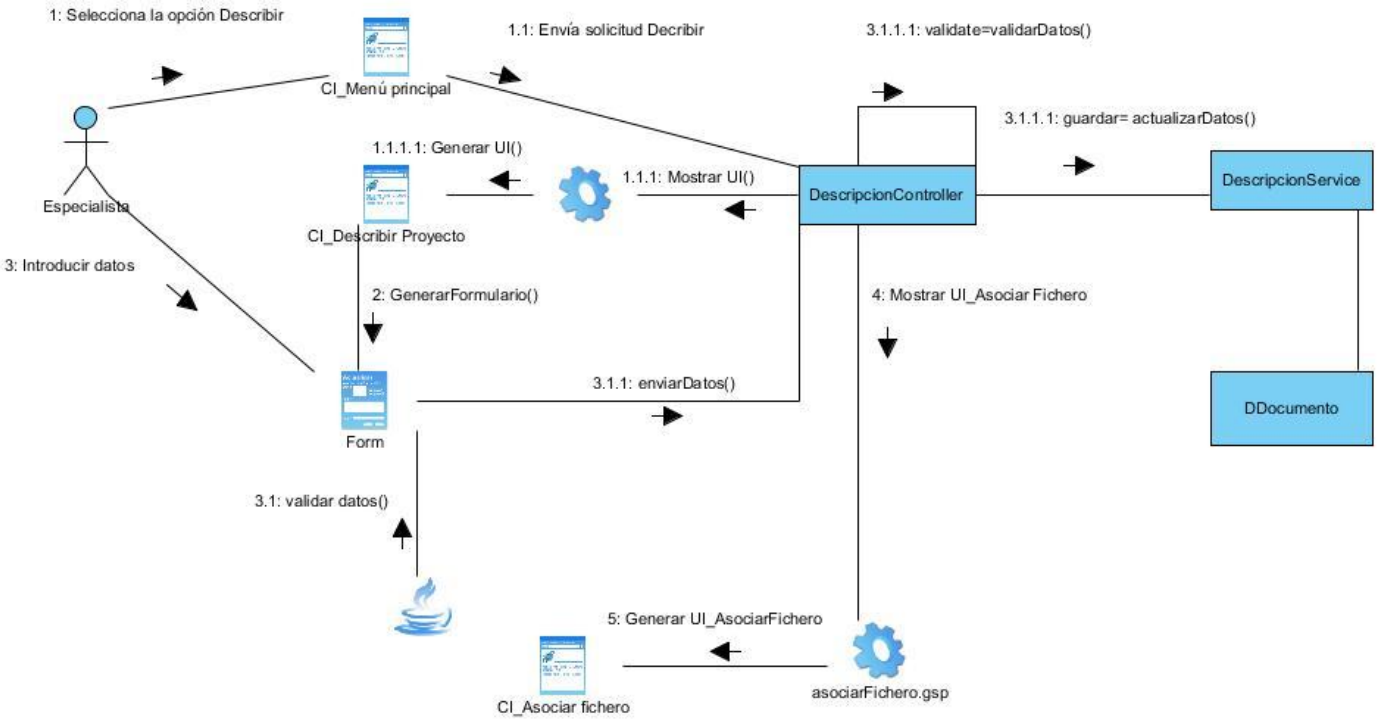


Figura 8 Diagrama de Comunicación CU Describir proyecto

3.3.5 Modelo de datos

El modelo de datos constituye un conjunto de herramientas conceptuales para describir la representación de la información en términos de datos. Comprenden aspectos relacionados con: estructuras y tipos de datos, operaciones y restricciones (36). La tabla siguiente representa una parte del modelo de datos

conceptual que describe las entidades del módulo Procesamiento con sus atributos y las descripciones correspondientes.

Entidad	**DDocumento**	
Descripción	Recoge los datos de los proyectos.	
Atributo	**Tipo**	**Descripción**
nombreCorto	String	Almacena el nombre corto del sistema. No puede ser nulo
objetivoGeneral	String	Almacena el objetivo general. No puede ser nulo
alcance	String	Almacena el alcance. No puede ser nulo
impacto	Integer	Almacena impacto del proyecto
entidadDesarrolladora	NEntidadDesarrolladora	Almacena la entidad desarrolladora del proyecto. No puede ser nulo
fechaInicio	Date	Almacena la fecha de inicio del proyecto. No puede ser nulo
fechaFin	Date	Almacena la fecha de fin del proyecto. No puede ser nulo
tipoArtefacto	NArtefacto	Almacena el tipo de artefacto
numeroRegistro	String	Almacena el número de registro del proyecto en el Centro Nacional de Derecho de Autor (CENDA)
clasificacionCliente	NClasificacionCliente	Almacena la clasificación del cliente
clasificacionNaturaleza	NClasificacionNaturaleza	Almacena la clasificación de la naturaleza del cliente
cliente	NCliente	No puede ser nulo
estadoCompletitud	Integer	Almacena el estado de completitud del proyecto
resumenTecnico	String	Almacena el resumen técnico del proyecto

comentariosValoraciones	String	Almacena las valoraciones asociadas al proyecto
fechaCreacion	Date	Registra la fecha en la que se crea la entidad en Base de Dato. No puede ser nulo
unidadConservacion	DUnidadConservacion	Representa la ubicación física del proyecto. No puede ser nulo
notificado	boolean	Representa si se realizó la notificación del sistema luego de pasados dos años de registrado el proyecto.

Tabla 7 Descripción de la entidad DDocumeto

Entidad	**DFichero**	
Descripción	La entidad se utiliza para registrar los ficheros asociados a los documentos	
Atributo	**Tipo**	**Descripción**
nombre	String	Nombre del fichero
extension	String	Extensión del fichero
tamanno	float	Tamaño en MB del fichero
documento_id	String	Llave foránea del documento en la entidad fichero

Tabla 8 Descripción de la entidad DFichero

Entidad	**DProductor**	
Descripción	Se utiliza para registrar los datos del autor del documento	
Atributo	**Tipo**	**Descripción**
formaAutorizadaNombre	String	Nombre del productor
resennaHistorica	String	Breve descripción del autor
rol	NRolIngenieria	Almacena el rol del autor en el proyecto

Tabla 9 Descripción de la entidad DProductor

Entidad	**DNivel**	
Descripción	Representa un nodo del cuadro de clasificación	
Atributo	**Tipo**	**Descripción**
Tipo_id	String	Representa la llave foránea relativa a la entidad tipo_id
directorio	String	Almacena la ubicación lógica, el nivel
acronimo	String	Almacena el acrónimo del nivel

Tabla 10 Descripción de la entidad DNivel

3.4 Conclusiones parciales

En este capítulo se realizó el análisis y diseño del sistema a partir de la modelación de algunos de sus artefactos para el caso de uso seleccionado, por ejemplo el diagrama de clases del Análisis y diagrama de interacción que sirven de guía para la implementación de la solución propuesta. Se describió la arquitectura de diseño del software que permite que se puedan realizar cambios en el código fuente, según las necesidades de los clientes, sin que se afecten las capas no involucradas; y también se definieron los patrones utilizados en el diseño, los cuales brindan una mejor comprensión en el desarrollo de los procesos de negocio. También se representó una vista de una parte del modelo de datos conceptual donde se describió las entidades del módulo Procesamiento con sus atributos y las descripciones correspondientes.

Capítulo 4. Implementación y prueba

4.1 Introducción

En el siguiente y último capítulo de esta investigación se desarrolla la implementación y prueba del sistema, a partir de los resultados del análisis y diseño. Como parte de la implementación se muestra, el diagrama de componentes, que representa la vista estática y dinámica del sistema y muestra la organización y dependencias entre un conjunto de componentes; y el diagrama de despliegue que modela el hardware utilizado en las implementaciones del sistema y las relaciones entre sus componentes. Se definen las pruebas de software que garantiza la calidad y revisión final del cumplimiento de los requisitos y especificaciones de diseño.

4.2 Implementación

La implementación comienza con el resultado del diseño e implementa el sistema en términos de componentes, siendo su propósito fundamental el desarrollar la arquitectura y el sistema como un todo. Su resultado principal es el modelo de implementación el cual incluye subsistemas de implementación y sus dependencias, componentes y la vista de la arquitectura de dicho modelo. El modelo de implementación constituye la entrada principal de la etapa de pruebas (13).

4.2.1 Diagrama de Componentes

Un diagrama de componentes muestra la organización y las dependencias entre un conjunto de componentes físicos de un sistema. Para todo sistema se han de construir una serie de diagramas que modelan tanto la parte estática, como dinámica, pero llegado el momento se debe materializar en un sistema implementado que utilizará partes ya implementadas de otros sistemas, y es justo esto lo que pretendemos modelar con los diagramas de componentes. Viéndolo así, un diagrama de componentes se considera un tipo especial de diagrama de clases que se centra en los componentes físicos del sistema (37).

A continuación se muestra una vista general de la composición del sistema Xabal Arkheia mediante la representación del diagrama de componentes teniendo en cuenta la estructura de implementación de la aplicación, este a su vez muestra la relación con otros componentes que representan los módulos que conforman el sistema y los plugins que usa. También se muestra la interacción con la base de datos del sistema, la base de datos del Gespro y con la interfaz del sistema eXcriba.

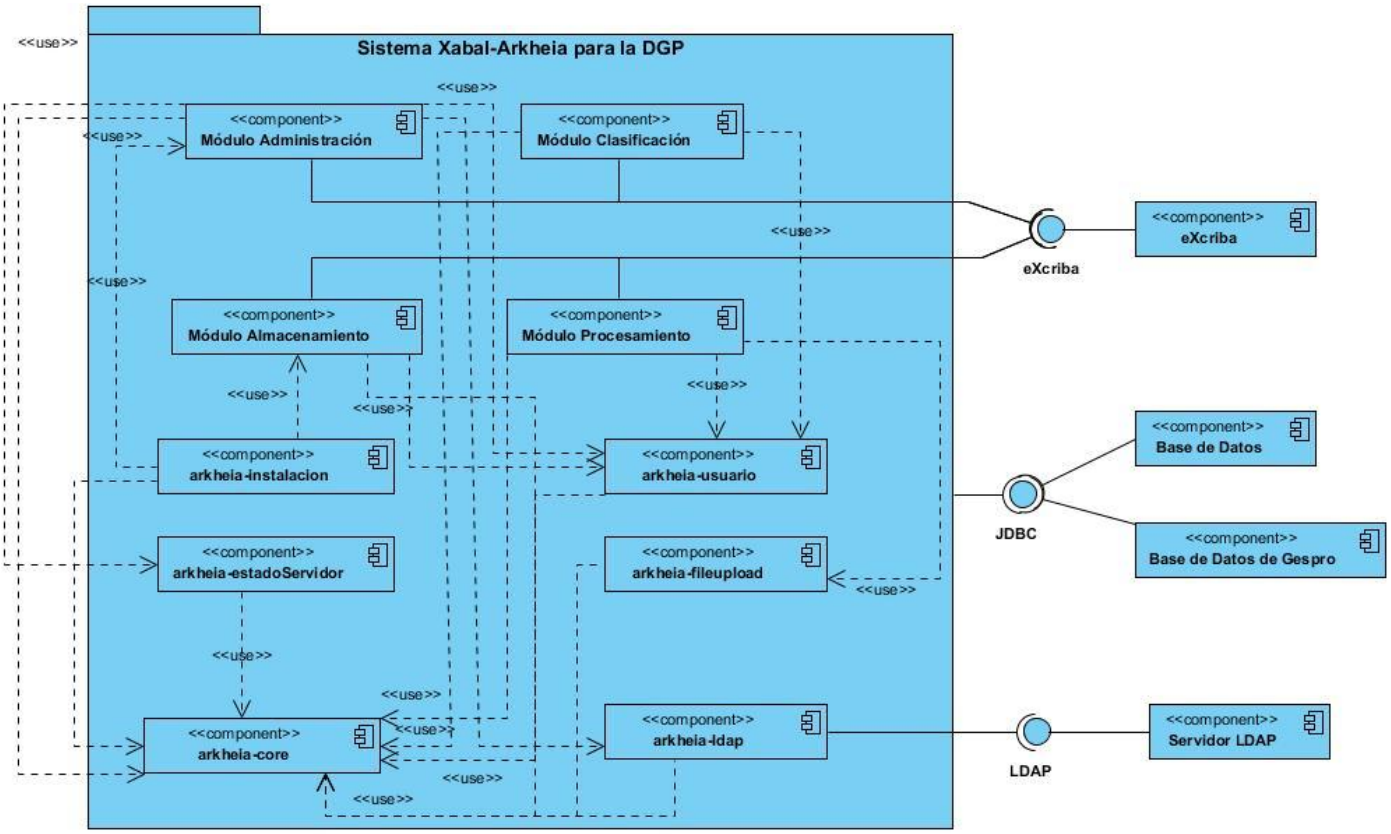


Figura 9 Diagrama de Componentes del Sistema Xabal - Arkheia para la DGP

La siguiente figura muestra el diagrama de componentes del módulo Procesamiento en el que se representa como componente principal implementado el módulo Procesamiento integrado por los componentes Views, Controllers, Services y Domain, y la interacción entre ellos. Las views (vistas), contienen las vistas o interfaces de usuario y son responsables de mostrar las distintas acciones disponibles, en el framework Grails son representadas por GSP; en el caso de este módulo las vistas se agrupan como un conjunto de elementos en los componentes buscador, descripción, explorarCuadroClasificacion y ficheros. El componente controllers (controladores) se encarga de recibir los eventos desde las vistas y determinar que vista debe mostrarse, por ejemplo en este diagrama se representan los controladores BuscadorController.groovy, DescripcionController.groovy y también la relación con los commands que son componentes que utilizan los controladores que facilitan el enlace de datos, conversión y su validación. El componente Services (servicios) su función está en implementar la lógica de negocio; por lo general cada controlador implementa su propio servicio. Los servicios del módulo Procesamiento lo conforman los

componentes BuscadorService.groovy, DescripcionService.groovy, ExplorarCuadroClasificacionService.groovy y FicherosService.groovy. El componente Domain (Dominio), encargado de gestionar los datos de la aplicación, es donde se encuentra las entidades o clases que persisten en la Base de Datos. El módulo Procesamiento por ejemple contiene las entidades DDocumento, DFichero, DProductor y DNivel. Ver Anexo 4 para consultar el diagrama de componentes del módulo Clasificación.

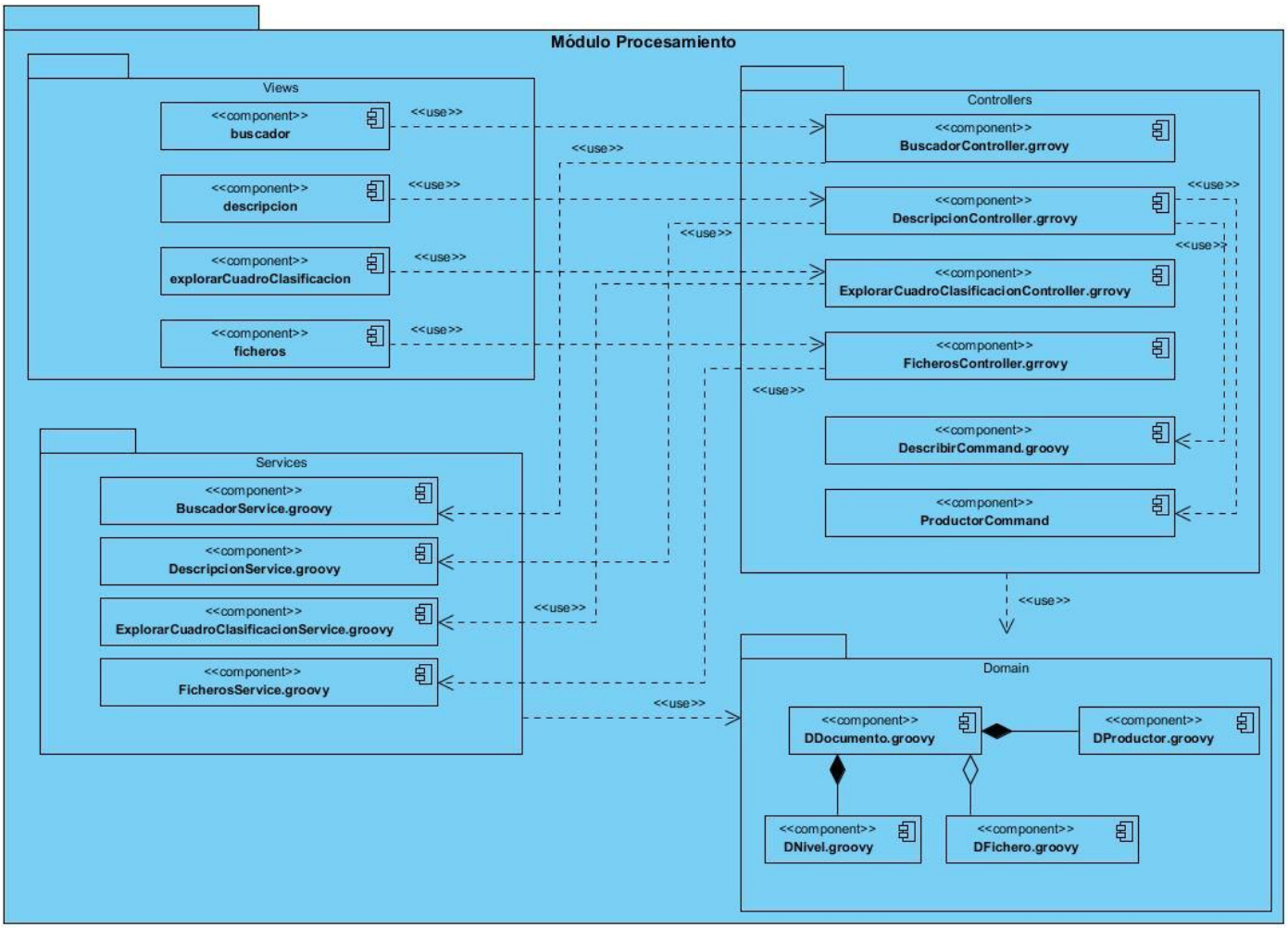


Figura 10 Diagrama de Componentes del módulo Procesamiento

4.2.2 Descripción de Componentes

Notificación de los documentos con dos años de archivados

Esta funcionalidad permite que transcurrido dos años luego de registrar el proyecto, el sistema debe lanzar una notificación del tiempo que lleva registrado el software, la misma constituye un requerimiento del cliente. A partir de este tiempo el sistema sigue registrando el tiempo de archivado el proyecto, aunque no notifica.

Para dar cumplimiento al *RF64. Emitir alerta una vez cumplido dos años a partir de la fecha de ser archivado,* el equipo de desarrollo se apoyó en el atributo *dateCreated* de todas las clases de dominio de los sistemas realizados con la arquitectura del Xabal Arkheia (38). Este atributo toma valor automáticamente en el momento de inserción en base de datos. Con el objetivo de marcar un documento que ya haya sido notificado, se le agregó a la clase DDocumento un atributo booleano.

Teniendo en cuenta estas características, se pueden consultar los documentos con 2 años a partir de la fecha de ser archivado con la siguiente consulta:

```
List<DDocumento> lista = DDocumento.findAllByDateCreatedLessThanEqualsAndNotificado((new Date() - 730), false)
```

La lista resultante constituyen los nuevos proyectos que deben notificarse según el requisito del tiempo de registro en el sistema. Por cada proyecto existente en la misma, se debe marcar como notificado y enviarles a los administradores del sistema una notificación por correo. El siguiente fragmento de código ilustra esta idea:

```
for (DDocumento documento in lista) {
    mensajeService = new MensajeService()
    documento.notificado = true
    if (documento.save(flush: true)) {
        def correos = DUsuarioSistema.findAllByAdmin(true)*.correo
        if (correos) {
            mensajeService.enviarMensajeCorreo(
                    correos as String[],
                    "Notificación de caducidad de documentos",
                    """Estimado administrador:
El proyecto con nombre ${documento.nombre}, identificado bajo el nombre corto ${documento.nombreCorto}, lleva dos años
```

Figura 11 Representación de la funcionalidad notificarCaducidad

Estas líneas de código se encuentran en el componente Procesamiento formando parte del método *notificarCaducidad()* y debe ejecutarse diariamente. Para lograrlo, al iniciar el sistema se crea una tarea que cada 24 horas invoca al método de notificación de caducidad.

De igual forma, se le da solución al *RF63. Cargar datos de los proyectos archivados en el Gespro según entidad desarrolladora*. A continuación se explica el proceder para con este requisito.

Integración con el Gespro

El Gespro almacena los proyectos terminados un una base de datos a la que se le creó un usuario para la personalización del Arkheia, con permisos de lectura. Constituye un requerimiento esencial que se carguen los datos desde el Gespro, todos los sábados a las 11:00 de la noche. A esta parte de este requisito se le puede dar respuesta utilizando las funcionalidades del plugin *Quartz para Grails* (39). Solo bastaría configurar un Job o Trabajo para que se ejecute a la hora deseada. Para elaborar la expresión de cron, la autora del documento se apoyó en el *Tutorial de Cron* (40). El código que se muestra a continuación muestra el uso del plugin para programar la ejecución de la extracción de los datos del Gespro:

```
class RetrieveDataToArchiveJob {
    def descripcionService

    static triggers = {
        cron name: 'RetrieveDataToArchiveJobTrigger', cronExpression: "0 0 23 ? * 7 *"
    }

    def execute(){
        descripcionService.retrieveProjectsToArchive()
    }
}
```

Figura 12 Definición del trabajo asociado a la recolección de datos del Gespro

La implementación de la extracción de datos del Gespro está definida en el método *retrieveProjectsToArchive()* del servicio *DescripcionService* del componente Procesamiento. La primera vez que se ejecuta este método se extraen todos los datos hacia la base de datos del Arkheia y se almacena la fecha en la que se ejecutó la consulta de la extracción. Para la próxima ejecución del método se obtendrían todos los proyectos terminados entre la fecha de la última ejecución y el momento en que se ejecuta el método. Con la ejecución de este método también se le da cumplimiento al *RF62. Actualizar los nomencladores desde el GESPRO*, donde se realiza una actualización de los valores de los nomencladores Cliente, Clasificación del Cliente, Naturaleza de la Clasificación y Entidad Desarrolladora.

Subir ficheros grandes al sistema

El framework Grails implementa la subida de ficheros al servidor atendiendo las peticiones de tipo *multipart/form-data* con la clase *MultipartFile* del framework Spring, manteniendo todos los ficheros en memoria. Esta forma tiene la desventaja de que si se subiesen al servidor ficheros grandes, puede provocar

un exceso de consumo de la memoria en el servidor. Este comportamiento puede deshabilitarse en el fichero de configuración *grails-app/conf/Config.groovy,* modificando la entrada dedicada al manejo de ficheros de la siguiente forma:

```
// configure passing read-only to OSIV session by defaul
grails.hibernate.osiv.readonly = false

menuFile = "menu.properties"
//----------------------Manually handle large file upload
grails.web.disable.multipart = true // deshabilita el us
//grails.disableCommonsMultipart true

grails.databinding.dateFormats = ['dd/MM/yyyy', 'MM/dd/y
environments {
```

Figura 13 Modificación del archivo de configuración de la aplicación para deshabilitar el manejo de la subida de ficheros

El siguiente paso es la introducción del uso de la clase *DiskFileItem* del paquete *fileupload* de las funcionalidades comunes de Apache. La implementación del manejo de la subida de ficheros con esta clase, permite definir la ubicación final del fichero a subir, la cantidad de memoria máxima a consumir y el tamaño máximo de los fragmentos de ficheros con los que trabajará el servidor. La siguiente imagen muestra el método adicionado para manejar esta funcionalidad:

```
List<DiskFileItem> getRequestItems(HttpServletRequest request, Map configuracion) throws Exception {
    List<DiskFileItem> items = []
    if (ServletFileUpload.isMultipartContent(request)) {

        DiskFileItemFactory factory = new DiskFileItemFactory(); // Create a factory for disk-based file items
        if (configuracion?.directorioTemporal) {
            factory.setRepository(configuracion.directorioTemporal);
        }
        if (configuracion?.memoriaMax) {
            factory.setSizeThreshold(configuracion.memoriaMax);
        }

        ServletFileUpload upload = new ServletFileUpload(factory); // Create a new file upload handler
        if (configuracion?.tamannoMax) {
            upload.setSizeMax(configuracion.tamannoMax);
        }
        // Parse the request
        items = upload.parseRequest(request)
    }
    return items
}
```

Figura 14 Implementación de la subida de ficheros al servidor

Este método se encuentra implementado en la clase *FileuploadService* del componente *arkheia-fileupload,* y recibe como parámetros la petición realizada y un mapa de valores con las configuraciones necesarias para su funcionamiento.

Integración con LDAP

La personalización del Arkheia para la DGP se desplegará en un servidor de la universidad. Constituye un requisito de estema el *RF72. Implementar autenticación por DNS*, por lo que se cambió la autenticación implementada en los sistemas Arkheia, soportada por el plugin de seguridad *spring-security-core* (41). Para garantizar la autenticación por DNS se hizo uso de un conjunto de clases que implementan este tipo de autenticación contra el LDAP UCI. La implementación de esta integración se encuentra en el controlador *LDAP* del componente Arkheia-LDAP en el método *autenticarUsuario* y debe garantizar el funcionamiento de la anterior forma de autenticación. A continuación se muestra un resumen de la implementación del mismo:

```
def autenticarUsuario(LoginCommand cmd){
    if (cmd.validate()){
        def usuario = DUsuarioSistema.findByUsername(cmd.j_username)    1
        if (usuario){
            if (usuario.deleted){...}
            if (!usuario.enabled) {...}
            if (usuario.passwordExpired) {...}          2
            if (usuario.accountLocked) {...}
        } else {...}

        def count = request.session.tryCounter;
        boolean doCaptcha = false
        if (count != null && count >= 3) {...}                  3
        boolean continuar = false
        if (count >= 3){...}else{continuar = true}
        if (!continuar){
            flash.error = message(code: "default.login.captcha.message")
            return redirect(action: "auth")
        }

        if (ldapService.autenticarUsuario(cmd)){            4
            if (usuario instanceof DUsuarioSistema) {
                int tiempo = configuracionService.tiempoContrasenna
                int dias = new Date() - usuario.fechaContrasenna      5
                if (!usuario.isAdmin()) {...}
            }
            arkheiaSecurityService.autenticarUsuario(cmd.j_username, cmd.j_password, request.requestedSessionId)
            return redirect (uri: "/")      6
        } else {
            flash.error = message(code: "springsecurity.errors.login.fail")
        }
    } else {...}
    redirect(action: "auth")
}
```

Figura 15 Representación de la funcionalidad autenticarUsuario

Cuando un usuario intenta acceder al sistema, se procede de la siguiente forma:

1. Se verifica si es usuario del sistema.
2. Si es un usuario del sistema, el sistema verifica que esté habilitado, que su cuenta no esté expirada, ni su contraseña y que no se encuentre bloqueado. Si el usuario incurre en uno de los casos, el sistema muestra la interfaz de autenticación con el mensaje para el caso específico.
3. Se verifica el mecanismo de defensa contra la autenticación por fuerza bruta.
4. Se autentica al usuario contra el servidor LDAP de la UCI o contra la base de datos del sistema.
5. Una vez autenticado se verifica que la contraseña del usuario no haya expirado.
6. El sistema muestra la interfaz principal del sistema.

Integración con el eXcriba

Como parte de la solución para la integración con el eXcriba, este sistema brindará tres servicios uno para la autenticación, otro para devolver las áreas (término equivalente a centros en Arkheia) y otra funcionalidad para que devuelva los proyectos.

4.2.3 Validaciones en el sistema

La validación es el proceso de comprobar que lo que se ha especificado es lo que el usuario realmente desea, es decir, se trata de evaluar el sistema o parte de este durante o al final del desarrollo para determinar si satisface los requisitos iniciales.

1. validación en el cliente,
2. validación en el servidor, y
3. tratamiento de excepciones.

Los parámetros de entrada fueron validados en el cliente –con JavaScript– y en el servidor –con Groovy o usando los propios *constraints* de las clases de dominio–, con el objetivo de garantizar la integridad de los datos procesados. Para la validación de los datos de entrada en el servidor se definieron clases servicio que serán invocados por el controlador que reciba estos parámetros. La nomenclatura de estas clases se compone por el *<nombre de la funcionalidad que estará validando>ValidatorService.groovy* e implementan la interfaz definida en la siguiente figura:

Los *constraints* constituyen la garantía de que los datos que se van a persistir no contienen errores. En la siguiente figura se muestra la definición del *constraints* de la clase del dominio correspondientes a los datos del documento:

```
static constraints = {
    nombre nullable: false, blank: false
    nombreCorto nullable: false, blank: false
    objetivoGeneral nullable: false, blank: false
    alcance nullable: false, blank: false
    impacto nullable: true, min: 0, max: 1000
    entidadDesarrolladora nullable: false
    fechaFin nullable: false
    fechaInicio nullable: false
    estadoCompletitud min: 0, max: 100, nullable: true
    tipoArtefacto nullable: true
    numeroRegistro(nullable: true, blank: true)
    usuarioRegistra nullable: true
    clasificaionCliente nullable: true
    clasificacionNaturaleza nullable: true
    productores minSize: 1, validator: { val, obj ->
        if (!val || val.size() == 0) {
            return ["descripcion.productores.empty.message"]
        }
    }
    fechaCreacion nullable: true
    unidadConservacion nullable: true
    resumenTecnico nullable: true, blank: true
    comentariosValoraciones nullable: true, blank: true
}

String toString() {
    nombre
}
```

Figura 16 Constraints de la clase DDocumento

Cuando el error es externo al módulo, se captura como una *Exception*, informándole al usuario de su ocurrencia. Si es una violación de las funcionalidades implementadas, entonces se le informa al usuario capturando y mostrando la excepción del módulo a través de la clase <nombre de la clase>*Exception.groovy.*

4.2.4 Diagrama de Despliegue

Los diagramas de despliegue se utilizan para modelar la vista de despliegue estática de un sistema. Esto implica modelar la topología del hardware sobre el que se ejecuta el sistema. Muestra las relaciones físicas entre los componentes hardware y software en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software (procesos y objetos que se ejecutan en ellos) (42).

El diagrama de despliegue está conformado por una PC cliente que se comunica mediante un navegador web a través del protocolo http con el Sistema Xabal Arkheia, desplegado en el Servidor de Aplicación que se conecta al Servidor de Base de Datos PostgreSQL y al Servidor de Base de Datos del Gespro mediante protocolo tcp/ip, y también al Servidor Web de eXcriba a través de http de donde se obtiene la documentación generada en los expedientes de proyecto.

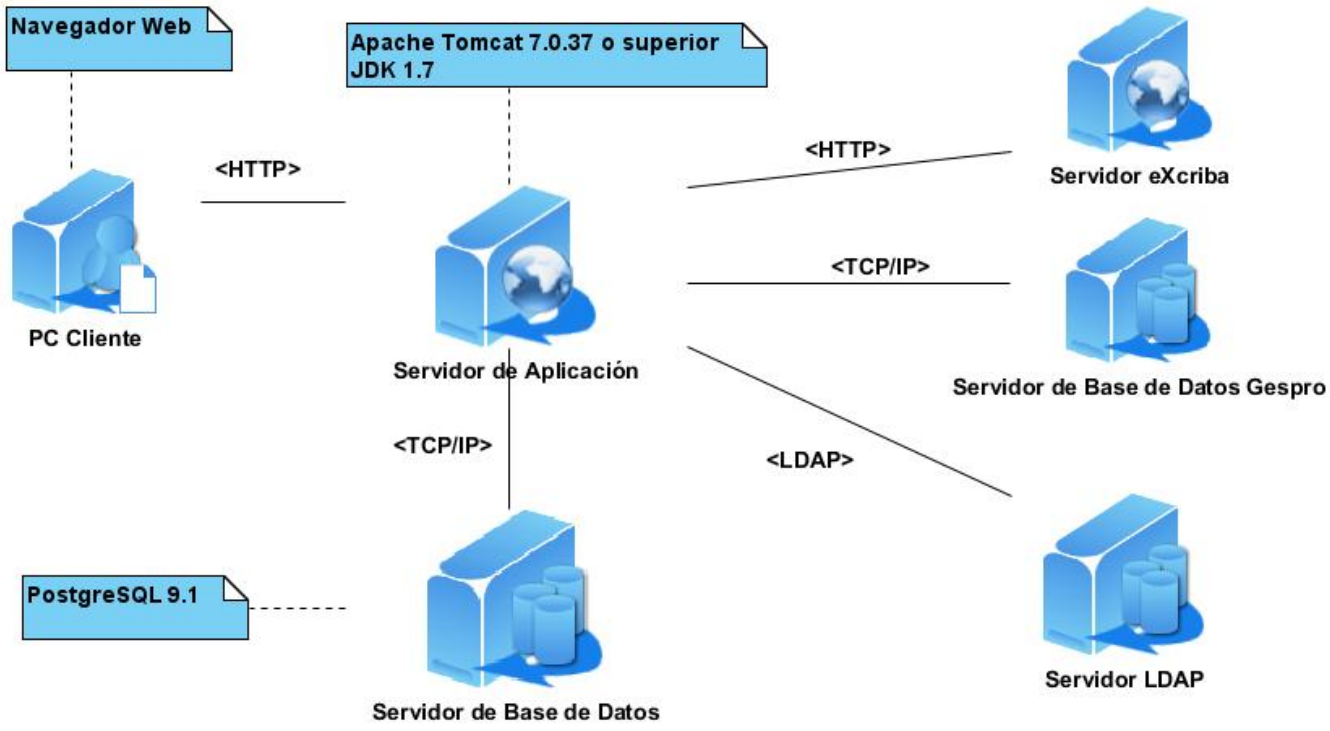


Figura 17 Diagrama de Despliegue del Sistema Xabal Arkheiapara la DGP.

A continuación se realiza una descripción de los nodos y artefactos que componen el diagrama de despliegue:

PC Cliente: destinada a las funciones de cualquier usuario.

Servidor de Aplicación: encargado de atender las peticiones de los usuarios y proporcionar la respuesta.

Servidor de Base de Datos: almacena toda la información que se gestiona en la institución.

Servidor eXcriba: brinda documentación generada en los expedientes de proyecto.

Servidor de Base de Datos Gespro: Base de Datos de donde se obtiene la información que provee el Gespro.

Servidor LDAP: Servidor que permite la autenticación por LDAP.

Los conectores están definidos por los protocolos de comunicación entre los artefactos y los nodos:

HTTP o Hypertext Transfer Protocol (en español Protocolo de transferencia de hipertexto): define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web (clientes, servidores, proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

TCP/IP (Protocolo de Control de Transmisión/Protocolo de Internet): es la base de Internet, y sirve para enlazar computadoras que utilizan diferentes sistemas operativos, incluyendo PC, minicomputadoras y computadoras centrales sobre redes de área local (LAN) y área extensa (WAN).

LDAP: (Lightweight Directory Access Protocol/Protocolo Ligero de Acceso a Directorios) es un protocolo de acceso unificado a un conjunto de información sobre una red.

4.3 Prueba

En el flujo de trabajo de la prueba verificamos el resultado de la implementación probando cada construcción. Sus objetivos van encaminados a planificar las pruebas necesarias en cada iteración, diseñar e implementar las pruebas creando los casos de prueba, los procedimientos de prueba y componentes de prueba ejecutables para automatizar las pruebas, y por último realizar las diferentes pruebas y manejar los resultados de cada una sistemáticamente. (13) En resumen el objetivo principal de esta etapa es diseñar pruebas que tengan la mayor probabilidad de encontrar el mayor número de errores con la mínima cantidad de esfuerzo y tiempo posible (32).

4.3.1 Método de Prueba: Pruebas de Caja Negra

La prueba de caja negra denominada también prueba de comportamiento, se centra en los requisitos funcionales del software, o sea permite obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y terminación (32).

4.3.2 Técnica de prueba

Dentro del método de Caja Negra la técnica de la *Partición de Equivalencia* permite examinar los valores válidos e inválidos de las entradas existentes en el software, descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la definición de casos de pruebas que descubran clases de errores (32).

Diseño de caso de prueba

Un caso de prueba detalla una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo la que ha de probarse (13). A continuación se presenta un resumen del diseño de caso de prueba del CU Describir Proyecto donde se muestran los diversos escenarios que se presentan, su descripción, la respuesta que brinda el sistema y el flujo central de acciones que se ejecutan. Para una especificación más detallada del mismo que incluye las variables a probar Ver Anexo 5

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1: Describir proyectos.	Se describen los documentos que se desean registrar.	Muestra un mensaje indicando que se han guardado los datos satisfactoriamente.	1. Seleccionar la opción “Describir“. 2. Seleccionar la opción "Asignar Ubicación Lógica". Nota: La ubicación lógica se asigna a través de un árbol de selección. Ver **DCP_Gestionar Ubicación Lógica** del módulo Clasificación. 3. Introducir el resto de los datos solicitados. 4. Seleccionar la opción “Aceptar”.

			5. Seleccionar los ficheros a asociar. **Ver CP-Asociar Ficheros.** 6. Seleccionar la opción “Aceptar”.
EC 1.2: Opción “Eliminar” lListado de autores.	Se elimina el autor seleccionado del listado de autores	Elimina el autor seleccionado y actualiza el listado de autores seleccionado.	1. Seleccionar la opción “Describir“. 2. Seleccionar el(los) autor(es) deseado(s). 3. Seleccionar la opción que permite eliminar el autor deseado del listado de autores seleccionado.
EC 1.3: Datos obligatorios vacíos.	Se muestra un mensaje de error indicando que no se han introducido los datos obligatorios.	Muestra un mensaje indicando que faltan datos por introducir y los señala.	1. Seleccionar la opción “Describir“. 2. Seleccionar la opción "Asignar Ubicación Lógica". Nota: La ubicación lógica se asigna a través de un árbol de selección. Ver **DCP_Gestionar Ubicación Lógica** del módulo Clasificación. 3. Introducir el resto de los datos solicitados. 4. Seleccionar la opción “Aceptar”.
EC 1.4: Modificar Ubicación Lógica.	Ver **DCP_Gestionar Ubicación Lógica, SC 3. Cambiar Ubicación Lógica (a partir del paso 4 del flujo central)**.	Muestra la interfaz de Cambiar Ubicación Lógica.	1. Seleccionar la opción “Describir“. 2. Seleccionar la opción "Asignar Ubicación Lógica". Nota: La ubicación lógica se asigna a través de un árbol de selección. Ver **DCP_Gestionar Ubicación Lógica** del módulo Clasificación. 3. Seleccionar la opción “Modificar Ubicación Lógica”.
EC 1.5: Ver Ubicación Lógica.	Ver **DCP_Gestionar Ubicación Lógica, SC 2. Ver Ubicación Lógica**.	Muestra la interfaz de Ver Ubicación Lógica.	1. Seleccionar la opción “Describir“. 2. Seleccionar la opción "Asignar Ubicación Lógica". Nota: La ubicación lógica se asigna a través de un árbol de selección. Ver

			DCP_Gestionar Ubicación Lógica del módulo Clasificación. 3. Seleccionar la opción “Ver Ubicación".

Tabla 11 Resumen del diseño de caso de prueba del CU Describir Proyecto.

4.3.3 Resultados

Este epígrafe está dedicado a la descripción de las pruebas realizadas al sistema y se muestran los resultados de las mismas. Luego de la implementación fueron diseñados los casos de prueba y se comenzó a probar la aplicación, para verificar el cumplimiento de los requisitos establecidos en los casos de uso. Se establecieron tres iteraciones de pruebas donde se encontraron irregularidades de tipo funcionalidad, validación, ortografía, formato, error de interfaz, redacción y no correspondencia con el diseño de caso de prueba; dichos errores detectados en los CU o del sistema son reconocidos como no conformidades (NC), y a partir de las mismas se ofrece una respuesta que puede ser Resuelta o No procede. La siguiente tabla y gráfico de barras correspondiente muestran un resumen de las no conformidades detectadas en cada iteración en las pruebas realizadas a la aplicación y su respuesta respectiva.

No. Iteraciones	Resuelta	No procede	Total NC
1ra iteración	44	7	51
2da iteración	6	19	25
3ra iteración	21	2	23

Tabla 12 Resultados de las pruebas de liberación aplicadas al sistema.

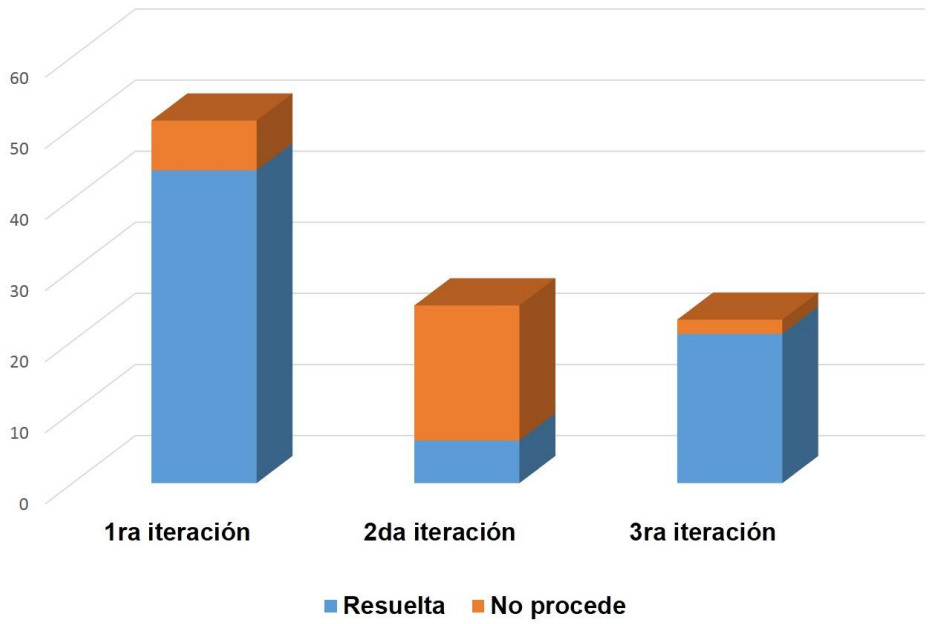


Gráfico 1 Resultados de las pruebas de liberación aplicadas al sistema.

4.4 Conclusiones Parciales

En este capítulo se muestra el desarrollo del sistema, evidenciado a través de los diagramas de componentes y su interacción con los otros componentes de la aplicación; además el diagrama de despliegue, que refleja la topología del hardware sobre el que se ejecuta el sistema. Para comprobar el correcto funcionamiento del sistema y el cumplimiento de los requisitos, se aplicaron pruebas de calidad de Caja Negra, a partir de las cuales resultó la aparición de no conformidades que pudieron ser eliminadas.

Conclusiones

Con la culminación de la presente investigación se ha logrado dar cumplimiento al objetivo trazado de personalizar el Sistema Xabal Arkheia para la DGP. Dicha solución brinda un sistema integrado para la organización y control de las producciones en la UCI, lo que permite agilizar el proceso de consulta y chequeo de los proyectos y conservar el documento software en un archivo histórico.

Se desarrolló una aplicación que permite el registro de los datos adaptados por la DGP para la descripción de un software, posibilita también almacenar el código y la documentación de un proyecto en un archivo de gran tamaño, gestiona la creación de los cuadros de clasificación de forma no lineal y se logró la integración del sistema desarrollado con los sistemas Gespro y eXcriba de los cuales se obtiene información de la gestión de los proyectos, que permite completar la descripción de los mismos.

Recomendaciones

Con el propósito de enriquecer la propuesta de solución desarrollada, se recomienda:

- Sugerir a los equipos de desarrollo del eXcriba y Gespro registrar los proyectos con un identificador único en ambos sistemas.
- Proponer implementación de servicio web en el Gespro donde se pueda consultar la existencia de nuevos proyectos terminados a partir de una fecha determinada.
- Sugerir al equipo de desarrollo del eXcriba que concluya la implementación de los servicios web que brindarán al Arkheia.
- Continuar el desarrollo de la investigación para perfeccionar e incrementar las funcionalidades de la aplicación.

Referencias Bibliográficas

1. **Laboratorio de Investigaciones en Gestión de Proyectos, UCI.** DGP - Dirección General de Producción. [En línea] 2014. [Citado el: 2 de mayo de 2015.] http://gespro.dgp.prod.uci.cu/.
2. **Alvarez de Zayas, Carlos.** *METODOLOGIA DE LA INVESTIGACION CIENTIFICA.* Santiago de Cuba : s.n., 1995.
3. **Dirección General y Estratégica. [En línea] http://html.rincondelvago.com/direccion-general-y-estrategica.html.**
4. Ferriol Marchena, Martha Marina , y otros. ***Manual de procedimientos para el tratamiento documental.*** **2008.**
5. Angulo Méndez, Sandra Marisol. Conservamos, Guía Técnica de Preservación en Bibliotecas. [En línea] http://www.bibliotecanacional.gov.co/revistas/index.php/Conservamos/article/view/249.
6. INTEGRACIÓN DE SISTEMAS . [En línea] http://www.zendos.es/soluciones-empresariales/integracion-de-sistemas/.
7. Laboratorio de Investigaciones en Gestión de Proyectos, UCI. Manual de Usuario para GESPRO 13.05 Tomo I y II. [En línea] http://gespro-help.prod.uci.cu/introduction/#problema-original-y-la-necesidad.
8. eXcriba, Equipo de desarrollo de. ***Manual de Usuario Interfaz Web del Gestor de Documentos Administrativos eXcriba v3.0.***
9. SEPG. Administración presupuestaria. [En línea] http://www.oficinavirtual.pap.minhap.gob.es/sitios/oficinavirtual/es-ES/CatalogoSistemasInformacion/Docunet/Paginas/QueEs.aspx.
10. Guía Soluciones TIC. [En línea] 2015. http://www.guiadesolucionestic.com/sistemas-de-informacion/gestion-documental/administracion-y-gestion-electronica-de-documentos.
11. Archivist´s Toolkit. [En línea] http://archiviststoolkit.org.
12. Roura Sixto, Leandro. CIGED_Arkheia-Red de Centros_Arquitectura_vista_de_entorno. [En línea] http://excriba.prod.uci.cu/proxy/alfresco//api/node/content/workspace/SpacesStore/325d646d-9bef-4e2a-9ee0-7dfde1683255/CIGED_Arkheia-Red%20de%20Centros_Arquitectura_vista_de_entorno.doc?a=true.
13. Jacobson, Ivar, Booch, Grady y Rumbaugh, James. ***El Proceso Unificado de Desarrollo de Software.*** **2000.**
14. Larman, Craig . ***UML y Patrones I. Introducción al análisis y diseño orientado a objetos.*** **México : s.n., 1999.**
15. White, Stephen A. y otros. ***Guía de referencia y modelado BPMN.*** **2009.**
16. Musciano, Chuck y Kemedy, Bill . ***HTML La Guía Completa.*** **México : s.n., 1999.**
17. Eguíluz Pérez, Javier . ***Introducción a JavaScript.*** **2009.**
18. Brito, Nacho. ***Manual de desarrollo web con Grails.*** **2009.**
19. David Marco. davidmarco.es. [En línea] http://www.davidmarco.es/articulo/introduccion-a-groovy-i.
20. Sánchez, Jorge. Manual de Java. [En línea] 2004. http://www.jorgesanchez.net/programacion/manuales/Java.pdf.
21. Bustos, Prof. Guillermo . ***Guía de Uso de la Herramienta CASE Visual Paradigm Standard Edition Versión 8.0.*** **mayo, 2010.**
22. Team, Spring. SpringSource. [En línea] 2015. http://www.springsource.com/developer/sts.
23. The TortoiseSVN team. TortoiseSVN. [En línea] 2004-2015. http://tortoisesvn.net.
24. RAMOS MARTIN, MARIA JESUS. Sistemas gestores de bases de datos. [En línea] http://www.mcgraw-hill.es/bcv/guide/capitulo/8448148797.pdf.
25. Group, The PostgreSQL Global Development. PostgresSQL. [En línea] 2015. http://www.postgresql.org/.
26. Foundation, The Apache Software. Apache Tomcat. [En línea] 1999-2015. [Citado el: 03 de marzo de 2015.] http://tomcat.apache.org/.
27. Alvarez, Miguel Angel. ***Manual de JQuery.***
28. Foundation., The jQuery. jQuery. [En línea] 2015. http://jquery.com/.
29. Tutorials Point. JDBC Tutorial. [En línea] 2015. http://www.tutorialspoint.com/jdbc/.
30. Rodríguez, Txema. Genveta:dev Desarrollo y Software. [En línea] 16 de 06 de 2012. http://www.genbetadev.com/frameworks/bootstrap.
31. Ceria, Santiago. ***Casos de Uso. Un Método Práctico para Explorar Requerimientos.***
32. Pressman, Roger S. ***Ingeniería de Software. Un enfoque práctico.*** **5ta Edición.**
33. Roura Sixto, Leandro. Portal del Desarrollador del Proyecto Archivo. [En línea] 2014. http://10.128.60.26/archivo/.
34. Venete, Adriana . ***Introducción a los Patrones de Arquitectura.*** **2010/2011.**

35. Otero Vidal, Mari Carmen. Diagramas De Interaccion. [En línea] http://datateca.unad.edu.co/contenidos/204023/Otero_M._s.f._._Diagramas_De_Interaccion.pdf.
36. Zorrilla Pantaleón, Marta E. Modelos de datos. [En línea] 2010-2011. http://personales.unican.es/zorrillm/BasesDatos/02%20-%20Modelos%20de%20datos%20ER-UML-relacional.pdf.
37. Ferré Grau, Xavier y Sánchez Segura, María Isabel. Desarrollo Orientado a Objetos con UML. [En línea] http://www.uv.mx/personal/maymendez/files/2011/05/umlTotal.pdf.
38. Proyecto Archivo. Portal del desarrollador del Proyecto Archivo. *La clase Entidad.* [En línea] 6 de noviembre de 2014. [Citado el: 10 de mayo de 2015.] http://10.128.60.26/archivo/?p=10.
39. The Grails Project. Quartz plugin for Grails . *Grails.* [En línea] [Citado el: 10 de junio de 2015.] https://grails.org/plugin/quartz.
40. Terracota, Inc. CronTrigger Tutorial. *Quartz. Job Scheduler.* [En línea] [Citado el: 10 de junio de 2015.] http://www.quartz-scheduler.org/documentation/quartz-1.x/tutorials/crontrigger.
41. The Grails Project. Spring Security Core Plugin. *Grails.* [En línea] junio de 2015. [Citado el: 10 de junio de 2015.] https://grails.org/plugin/spring-security-core.
42. DIAGRAMA DE OBJETOS, SECUENCIA Y DESPLIEGUE EN UML. [En línea] http://datateca.unad.edu.co/contenidos/204023/Wordpress.com._s.f._._Diagramas_de_Objeto_Secuencia_y_Despliegue_en_UML.pdf.
43. Hernández León, Rolando Alfredo y Coello, Sayda. *El proceso de investigación científica.* Ciudad de La Habana : Editorial Universitaria, 2011.
44. König, Dierk. *Groovy in Action.* 2006.
45. Murphey, Rebecca . Fundamentos de jQuery.Creative Commons Attribution-Share Alike. [En línea] http://librojquery.com/.

Anexos

Anexo 1. RnF de Hardware

El código y la documentación de un proyecto generan un archivo comprimido que puede llegar a los 2GB en tamaño, y se estima un promedio de registro de 2 proyectos al mes, por lo que en 1 año quedarían registrados un total de 24 proyectos con una capacidad de 48GB. Esta capacidad representa de la capacidad total estimada para la PC Servidor del cliente un 4.69%, lo que permite estimar la duración de la capacidad por 21 años aproximadamente.

Anexo 2. RnF Rendimiento

Apache JMeter es una herramienta ideal para realizar pruebas de rendimiento de aplicaciones web, puede ser utilizada para analizar y medir el desempeño de una variedad de servicios. En esta investigación usamos la versión Apache JMeter 2.3.1 con el objetivo de realizar un estudio del rendimiento de la aplicación, haciéndole un gran número de peticiones en un tiempo determinado y así revisar y valorar las respuestas. Las pruebas se le realizaron a una funcionalidad que inserta 100 tuplas en base de datos, partiendo de la conexión de 100 usuarios realizando 20 peticiones cada 2 segundos en un servidor de aplicaciones de 2 GB de RAM con sistema operativo Linux Mint 17, como muestra la siguiente imagen:

Figura 18 Pantalla de configuración para 100 hilos en la prueba con JMeter

Para 4800 muestras, se obtuvo un tiempo promedio de respuesta por debajo de 1500 milisegundos, 0% de error a 16,9 peticiones por segundo.

Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
2gb ram	4800	1409	1407	2328	47	4135	0,00%	16,9/sec	54,0
TOTAL	4800	1409	1407	2328	47	4135	0,00%	16,9/sec	54,0

Figura 19 Resultados de la prueba

En un ambiente real se realizó la prueba para la conexión de 30 usuarios que realizan 3 peticiones cada 3 segundos, los resultados obtenidos para 1530 muestras, fueron de un tiempo promedio de respuesta de 76 milisegundos, 0% de error a 8,3 peticiones por segundo.

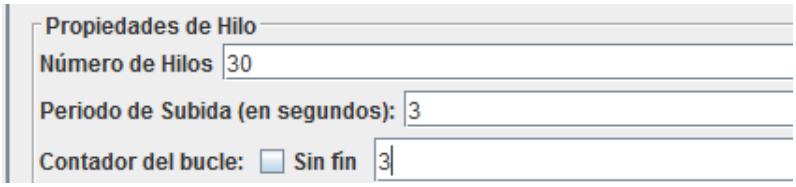


Figura 20 Pantalla de configuración para 30 hilos en la prueba con JMeter

Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
2gb ram	1530	76	78	94	31	297	0,00%	8,3/sec	26,7
TOTAL	1530	76	78	94	31	297	0,00%	8,3/sec	26,7

Figura 21 Resultados de la prueba

Anexo 3. Diagrama de clases del diseño CU Gestionar ficheros

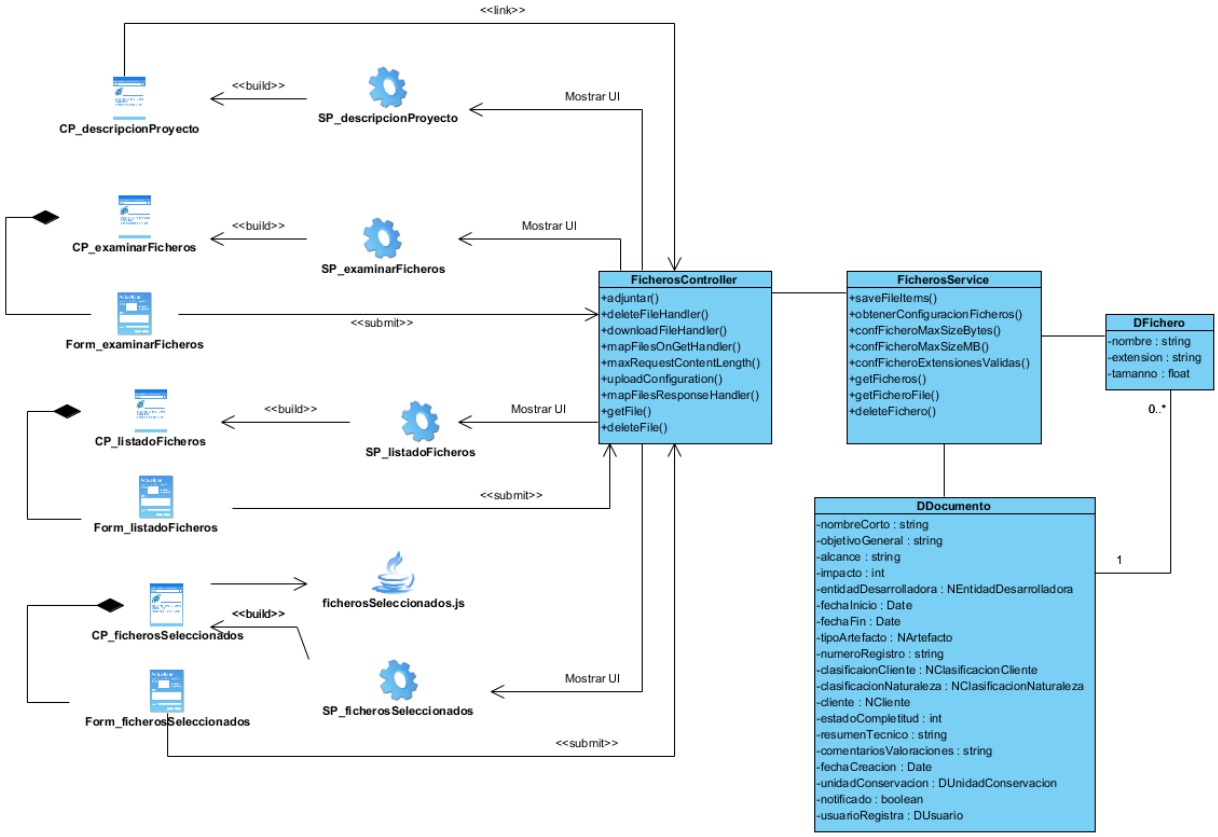
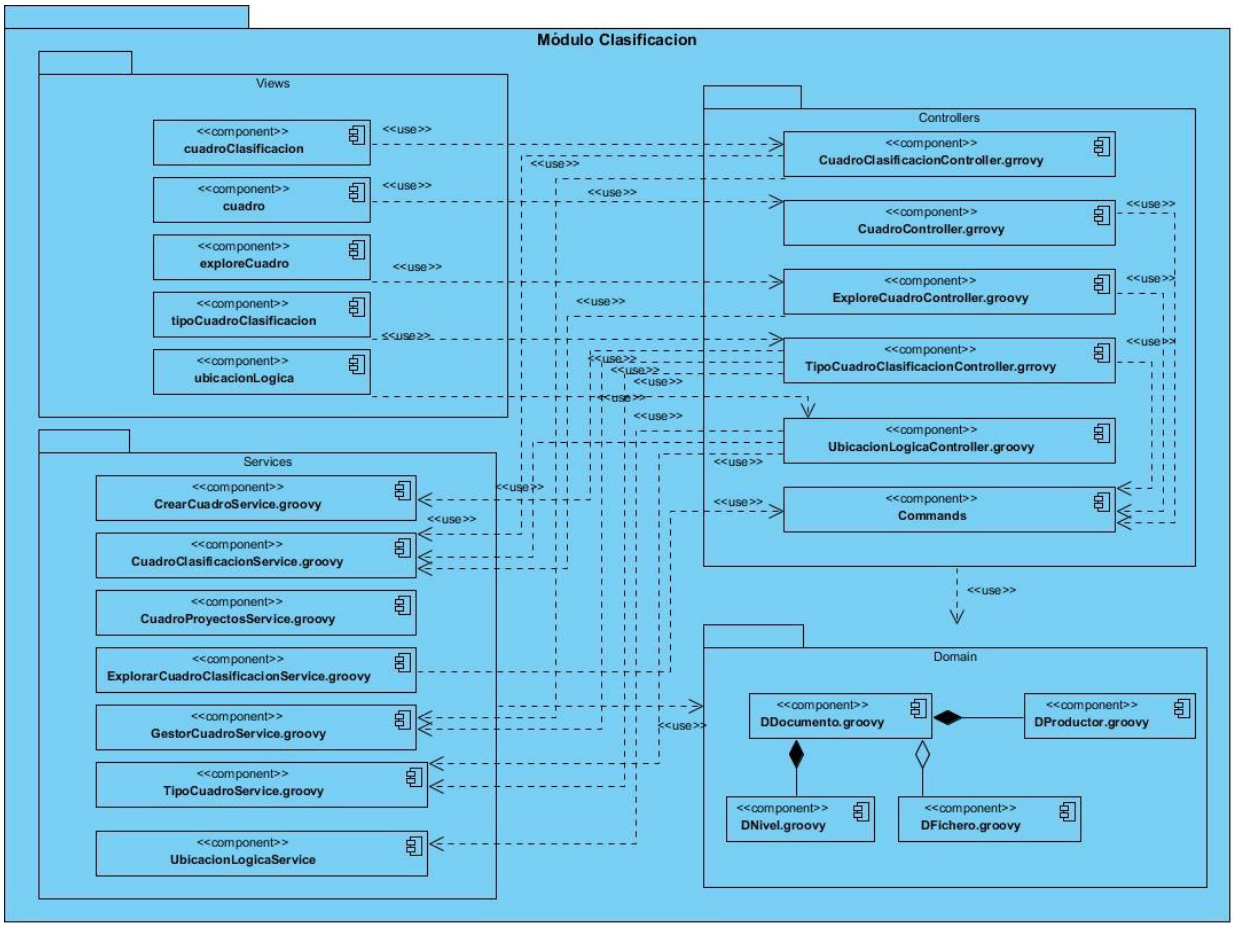


Figura 22 Diagrama de clases del diseño CU Gestionar ficheros

Anexo 4. Diagrama de Componentes del módulo Clasificación



Anexo 5. Vista del Caso de Prueba del CU Describir Proyecto con las variables a probar. Puede encontrar la descripción completa de este Caso de Prueba en el expediente de proyecto de la personalización del Xabal Arkheia para la DGP.