Sistema de Gestión Documental Open-Source

Tutor

Juanjo Prada

Responsables y Colaboradores

Pablo Andrade

Ian Coates

Pablo Acosta

Institución
ISA Ltda.

Integrantes
María Emilia Araujo
María Isabel Gabard Mutay
Diego Tacuabé Cabrera Altez

Instituto de Computación
Facultad de Ingeniería
Universidad de la República
2008 – 2009

Resumen

El proyecto Sistema de Gestión Documental Open Source, está enfocado al estudio y creación de un sistema electrónico que permite el manejo de distintos tipos de trámites dentro de una organización. Este sistema ofrece la gestión de dichos trámites manejándolos a través de flujos de trabajo entre distintas oficinas dentro de la organización, otorgando seguridad, confiabilidad y permitiendo en todo momento el rápido acceso a la información. El seguimiento que se puede realizar sobre los trámites mejora la gestión de la organización, permitiendo detectar problemas de funcionamiento. Estos trámites son basados en el decreto 500/991 [2] de la legislación uruguaya, y está enfocado principalmente hacia expedientes y formularios, que a su vez están formados por otros documentos. En el caso de expedientes, formados por una carátula y actuaciones y, en el caso de formularios, por sesiones.

Se realizó un importante análisis de la problemática contemplando que el producto final tiene que ser adaptable tanto para organizaciones públicas como privadas, cumpliendo con la legislación uruguaya.

Una etapa fundamental del proyecto abarcó la investigación de tecnologías que aporten a la solución de un sistema de Gestión Documental, basándose en herramientas open-source y estándares. Este profundo estudio contiene la comparación entre distintas alternativas para los diferentes puntos analizados, tales como, almacenamiento de datos y flujos de trabajo.

Se diseñó una arquitectura que permite extensibilidad por medio de un conjunto de bibliotecas básicas, asegurando una fácil incorporación de nuevos y diferentes trámites al sistema.

Al final del proyecto se realizó un prototipo que muestra el manejo e integración de las tecnologías elegidas en la investigación.
Dentro de las tecnologías utilizadas se encuentran:

- Java [19]
- XML [28]
- XPDL [29] – Worflows [25]
- MxGraph [16]
- JCR (Jackrabbit) [30]
- ZK [31]
- Postgres [39]
- Java Beans [32]
- Eclipse IDE [33]

Palabras Claves

Gestión documental, Trámite, Expediente, Formulario, Flujo de trabajo.

Índice

Capitulo 1 Introducción

1.1. Descripción general

Tanto las organizaciones públicas como privadas, realizan distintos trámites, para los cuales es necesario manejar grandes cantidades de documentos que son archivados, manteniendo los libros de registros de ubicación manualmente. Esto genera problemáticas como: lentitud, información no disponible o extraviada, robos, deterioros, documentos repetidos o en múltiples fuentes o faltantes, imposibilidad de acceso, dificultades de almacenamiento, entre otros. Con el objetivo de solucionar estos problemas se han implementado diferentes sistemas informáticos de Gestión Documental.

Algunas posibles definiciones sobre la Gestión Documental son las siguientes:

"La Gestión Documental es un conjunto de normas, técnicas y prácticas usadas para administrar el flujo de documentos de diferentes tipos en una organización, permitiendo la recuperación de información y asegurando la conservación de los documentos" [11].

"La Gestión Documental consiste en el uso de tecnologías y procedimientos que permiten una gestión eficaz y rentable y, un acceso unificado y seguro, a información generada y recibida en la organización." [12].

Hoy en día, los sistemas de Gestión Documental no solo registran los documentos que se encuentran en papel o en forma electrónica, sino que además controlan los flujos de trabajo y el proceso de tramitación, permitiendo agilizar los trámites administrativos e incrementar los controles sobre los archivos y documentos que circulan dentro de una organización. Por otro lado, también es posible la integración con distintos sistemas existentes en la organización, tales como sistemas contables.

Motivos por los que es necesario un Gestor Documental:

- Razones económicas: la no disponibilidad o localización de documentos por lo general es debido a que están siendo utilizados por otra persona, lo que provoca costos de personal, tiempo, recursos y dinero innecesarios.
- Razones informativas: evitar duplicar tareas; las buenas decisiones requieren buena información.
- Razones de seguridad: un documento que no puede ser visto por cualquier usuario estará marcado con algún sello de "Confidencialidad", pero esto no detiene al curioso.
- Razones jurídicas: necesidad de documentar las actuaciones realizadas sobre un trámite.
- Razones de conservación histórica: los documentos conforman la memoria colectiva.

Beneficios de la gestión documental:

- Racionalización de los recursos, pues se hace más fácil compartir la información, por lo tanto, se reducen situaciones como la duplicación de documentos archivados, fotocopias innecesarias, etc.
- Seguridad de la información, pues documentos de gran valor pueden estar custodiados y/o duplicados en diferentes lugares, mientras que se dispone para el uso diario de una copia.
- Facilidad y agilidad en la recuperación de información, reduciendo tiempo de consultas y tareas de archivo.
- Disminución de la masa documental, ahorro de espacio físico.
- Aumento de la efectividad y eficiencia administrativa originados por el rápido acceso a la información y su distribución, sin necesidad de trasladar documentos.

A continuación se definen algunas características que se deben tener en cuenta al momento de desarrollar un sistema de Gestión Documental:

- Almacenamiento: dónde almacenar los documentos, costos del almacenamiento, etc.
- Recuperación: tiempo de acceso a la información, búsquedas eficientes para encontrar los documentos necesarios, etc.
- Clasificación: cómo organizar los documentos, asegurar que los documentos estén archivados.
- Seguridad: evitar pérdidas de documentos, violación de la información o destrucción no deseada de documentos, ocultación de la información a quienes no la pueden ver.
- Custodia: qué documentos conservar y por cuánto tiempo, cómo se eliminan.
- Distribución: cómo y cuándo se distribuyen los datos, cuánto tiempo se puede tardar, etc.
- Flujo de Trabajo: manejo y reglas de flujo de los documentos, control sobre el flujo.
- Creación: control en el acceso para la creación y modificación por diferentes personas.
- Autenticación: cumplimiento de estándares para la autenticación, requisitos para su valoración legal, etc.

1.2. Objetivos y resultados esperados

Se plantearon desde el comienzo los siguientes objetivos para el proyecto:

Realizar completamente el análisis, especificación de requerimientos y diseño de un sistema de gestión de trámites basados en el decreto 500/991 [2] de la legislación uruguaya.

Estudiar distintas tecnologías que aporten a la solución realizando una evaluación y comparación de las distintas alternativas.

Diseño de una arquitectura completa y extensible.

Implementación de un prototipo que muestre el uso de las tecnologías elegidas y su integración.

A nivel tecnológico, la solución debe ser:

- Full web - independiente del navegador.
- Orientada a documentos basados en XML.
- Independiente del motor de base de datos.
- Independiente del sistema operativo.
- Orientada a servicios y utilización de BPM para la definición y flujos de procesos.
- Capaz de buscar full-text.
- Interoperable con sistemas similares.
- Desarrollada en Java utilizando tecnologías open-source.

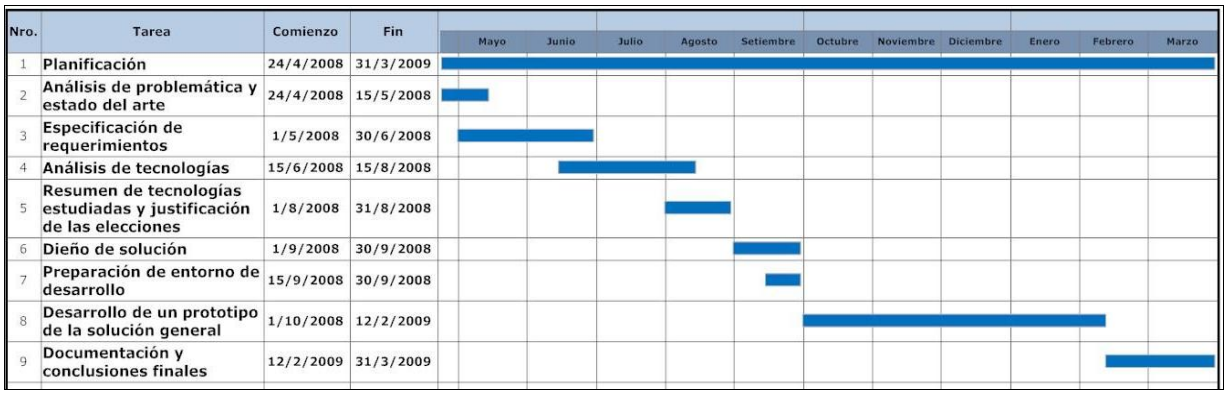
1.3. Planificación

El proyecto consistió en las siguientes fases:

1. *Planificación:* Al comienzo del proyecto se planificaron las diferentes fases y objetivos a cumplir en cada una de ellas.
2. *Análisis de la problemática y estado del arte:* En esta fase se estudió e investigó la problemática a resolver, adquiriendo un conocimiento de base y de aplicaciones similares ya existentes.
3. *Especificación de los requerimientos:* Se realizó el análisis y especificación de los requerimientos del sistema.
4. *Resumen de tecnologías estudiadas y justificación de las elecciones:* En esta etapa se realizó un estudio de las posibles herramientas que se podrían utilizar y que cumplen requisitos planteados al comienzo del proyecto. Sobre ellas se decidió justificando adecuadamente cuales utilizar en la implementación.
5. *Diseño de la Solución:* Se realizó el diseño completo para la solución final y se definió el alcance para la implementación.
6. *Preparación de entorno de desarrollo:* Se integraron las tecnologías a utilizar y se definió la estructura de la implementación del prototipo.
7. *Desarrollo de un prototipo de la solución general:* Se implementó y testeó la solución para el alcance fijado.
8. *Documentación y conclusiones finales:* Se preparó la documentación y presentación final del proyecto.

Las fases que requirieron documentación fueron finalizadas con la entrega del documento correspondiente. En este sentido se entregaron los documentos de las fases 2, 3, 4 y 5. Además se mantuvo actualizado un documento de seguimiento, donde se encuentran la planificación y las tareas realizadas a lo largo del proyecto.

En la figura 1 se muestra el Gantt del proyecto, describiendo el tiempo dedicado a cada fase del mismo.



Nro.	Tarea	Comienzo	Fin
1	Planificación	24/4/2008	31/3/2009
2	Análisis de problemática y estado del arte	24/4/2008	15/5/2008
3	Especificación de requerimientos	1/5/2008	30/6/2008
4	Análisis de tecnologías	15/6/2008	15/8/2008
5	Resumen de tecnologías estudiadas y justificación de las elecciones	1/8/2008	31/8/2008
6	Dieño de solución	1/9/2008	30/9/2008
7	Preparación de entorno de desarrollo	15/9/2008	30/9/2008
8	Desarrollo de un prototipo de la solución general	1/10/2008	12/2/2009
9	Documentación y conclusiones finales	12/2/2009	31/3/2009

1- Diagrama de Planificación del Proyecto

1.4. Estructura del documento

Este documento contiene los siguientes capítulos:

En el capítulo 2 se realiza el análisis de la problemática, en el se incluyen el resumen del marco legal en los que se basa el proyecto y el resumen de aplicaciones existentes, tanto en nuestro país como en el extranjero.

El capítulo 3 es sobre los requerimientos relevados y las herramientas investigadas. Aquí se describen brevemente los principales requerimientos para un sistema de Gestión Documental, junto con el resumen de las tecnologías investigadas y las conclusiones y elecciones de las herramientas a utilizar en una solución.

En el capítulo 4 se detalla la solución encontrada para el diseño y la arquitectura. Se muestran las vistas principales de la arquitectura del sistema y una descripción de la misma.

El capítulo 5 detalla el contenido del prototipo, su alcance y su implementación.

Los resultados obtenidos en el proyecto son mostrados en el capítulo 6 y en el capítulo 7 se muestran las conclusiones a las que se llegaron y el trabajo futuro que se puede realizar a partir de éste.

En los anexos se profundiza sobre los requerimientos relevados, las tecnologías investigadas y el diseño realizado. También se agrega un manual de usuario.

Capítulo 2 Análisis de la problemática

2.1 Introducción

En este capítulo se analizará, en la sección 2.2, la legislación uruguaya, sobre la que se basa el sistema de Gestión Documental para definir los trámites. Luego, en la sección 2.3 se estudiará el estado del arte, mostrando algunas de las soluciones que existen actualmente para un gestor documental.

2.2 Marco legal

La Legislación uruguaya unifica, por medio del Decreto Nº 500/991 [2], criterios para el manejo de tramitaciones dentro de las organizaciones tanto públicas como privadas. En este decreto se especifican algunos de los trámites que se desean implementar. Entre ellos se encuentra la Forma Documental *expediente*, los *formularios* y las *comunicaciones*. Dentro de estas últimas se tienen distintos tipos, *Oficio*, *Memorando*, *Circular* y *Carta,* que son utilizados para el manejo de todos los pedidos e intercambios de información entre las oficinas.

A continuación se describen brevemente cada uno de estos trámites.

Expediente

> Recopilación de documentos, formado para el tratamiento de aquellos asuntos que requieran mantener reunidas todas las actuaciones para tomar una resolución al respecto.
>
> Se formarán siguiendo el ordenamiento regular de los documentos que lo integra, en forma sucesiva y por orden de fechas.
>
> No se formará expediente con aquellos documentos que por su naturaleza no tengan relación directa con un acto administrativo ni lo hagan necesario, ni sea de ellos menester para la sustanciación de un trámite.
>
> Especialmente quedan comprendidos en esta prohibición las cartas, las circulares y los memorandos.
>
> Tampoco se formará expediente con aquellos asuntos que se tramiten exclusivamente a través de formularios.
>
> Los expedientes se identificarán por su número correlativo anual único para todo el organismo, el que será asignado por la unidad de administración documental.

El jerarca de cada dependencia o repartición fijará, dentro del plazo de ciento ochenta días a partir de la entrada en vigencia del presente decreto, la secuencia de las unidades administrativas que habitualmente deban participar en la sustanciación de cada tipo o clase de expediente por razón de materia, con la que se elaborará la correspondiente hoja de tramitación. Dicha hoja será puesta por la unidad de administración documental como foja inicial del expediente, a continuación de la carátula y antes de toda actuación.

Formularios

Los formularios son trámites de información estructurada que se utilizan de forma reiterada.

Especialmente se emplearán formularios para:

a) las gestiones de los funcionarios y la formulación de las documentaciones técnicas o administrativas rutinarias (licencias, solicitud de materiales, partes de personal, control de vehículos, control de documentos, informes de avance de obras, etc.);
b) las gestiones de los particulares relativas a prestaciones de servicios, cumplimiento de exigencias legales o reglamentarias (certificaciones, inscripciones, etc.), permisos, autorizaciones y otros actos de trámite directo o inmediato entre las dependencias competentes y los administrados.

Los formularios se individualizarán por su denominación, código identificatorio de la unidad emisora y número correlativo anual asignado por la unidad que centralice el sistema de formularios o, en su defecto, por la dependencia emisora.

Los formularios no requerirán carta o memorando de presentación ni expediente para su tramitación.

Oficio

El oficio será el documento utilizado cuando el órgano actuante debe dar conocimiento de sus resoluciones a otro órgano o formularle alguna petición para el cumplimiento de diligencias del procedimiento. Será objeto de numeración y registro por parte de la respectiva unidad de administración documental.

Circular

La circular será el documento utilizado para poner en conocimiento de los funcionarios órdenes o instrucciones de servicio, así como noticias o informaciones de carácter general. Se identificarán a través de un número correlativo anual asignado por la unidad emisora y se archivarán en la correspondiente unidad de administración documental.

Memorando

> El memorando se empleará para las instrucciones y comunicaciones directas del jerarca a un subordinado, o para la producción de información del subordinado a su jerarca, o para la comunicación en general entre las unidades. Los memorando se identificarán por un número correlativo anual asignado por el emisor. El receptor guardará el original y la copia de la contestación que hubiere emitido en forma escrita o a través de otro medio de comunicación.

Carta

> Toda otra comunicación escrita no contemplada en las anteriores.

Lo anterior es un breve resumen del decreto, destacando algunas de las menciones realizadas sobre las distintas Formas Documentales.

Normativa

Lo que sigue es la enumeración de una serie de leyes y decretos que hacen a la reglamentación del uso de los medios informáticos para el manejo de las distintas Formas Documentales.

La Ley 16736 del año 1996 [2], hace legítimas a las actuaciones administrativas realizadas por medios informáticos, con idéntica validez jurídica y valor probatorio que las actuaciones que se transmiten por medios convencionales. Lo que se muestra en los siguientes artículos.

> "Las administraciones públicas impulsarán el empleo y aplicación de medios informáticos y telemáticos para el desarrollo de sus actividades y el ejercicio de sus competencias, garantizando a los administrados el pleno acceso a las informaciones de su interés." (Artículo 694 Ley 16736).

> "Los trámites y actuaciones que conforman el procedimiento administrativo así como los actos administrativos podrán realizarse por medios informáticos. Su validez jurídica y valor probatorio serán idénticos a los de las actuaciones administrativas que se tramiten por medios convencionales. La firma autógrafa podrá ser sustituida por contraseñas o signos informáticos adecuados." (Artículo 695 Ley 16736).

> "La notificación personal de los trámites y actos administrativos podrá realizarse válidamente por correo electrónico u otros medios informáticos o telemáticos, los cuales tendrán plena validez a todos los efectos siempre que proporcionen seguridad en cuanto a la efectiva realización de la diligencia y a su fecha." (Artículo 696 Ley 16736).

> "La documentación emergente de la transmisión por medios informáticos o telemáticos constituirá de por sí documentación auténtica y hará plena fe, a todos sus efectos, en cuanto a la existencia del original transmitido. El que voluntariamente transmitiere un texto del que resulte un documento infiel, adultere o destruya un documento almacenado en soporte magnético, o su respaldo, incurrirá en los delitos previstos por los artículos 236 a 239 del Código Penal, según corresponda." (Artículo 697 Ley 16736).

Mediante la Ley 17243 del año 2000 [2], en la sección 3, referente al sistema informático del estado, se reglamenta el uso de expedientes electrónicos y de la firma electrónica y la firma digital.

> "El Estado, los Entes Autónomos y los Servicios Descentralizados deberán implantar el expediente electrónico para la sustanciación de todas las actuaciones administrativas. A tal efecto dispondrán los actos jurídicos y operaciones materiales tendientes al cumplimiento de esta norma en el menor tiempo posible, dando cuenta a la Asamblea General.
>
> El expediente electrónico es la serie ordenada de documentos registrados por vía informática, provenientes de la Administración o de terceros, tendientes a la formación de la voluntad administrativa en un asunto determinado, teniendo la misma validez jurídica y probatoria que el expediente tradicional." (Artículo 24 Ley 17243)
>
> "Autorizase en todo caso la firma electrónica y la firma digital, las que tendrán idéntica validez y eficacia a la firma autógrafa, siempre que estén debidamente autenticadas por claves u otros procedimientos seguros, de acuerdo a la tecnología informática." (Artículo 25 Ley 17243)

A continuación, se describe como la legislación uruguaya define y reglamenta la implementación de medios electrónicos de transmisión, almacenamiento y manejo de documentos en la Administración Pública.

<u>Documentos electrónicos:</u>

> "Los documentos electrónicos son conjuntos de datos digitales generados ya sea por vía de software específico o de software de gran divulgación" (Considerando VII - Decreto 83/001).
>
> "Constituirán instrumentos públicos, aquellos creados por medios informáticos que aseguren su inalterabilidad y sean redactados o extendidos por funcionarios competentes, según las formas requeridas y dentro del límite de sus atribuciones." (Artículo 1 Decreto 65/998)

Expediente Electrónico:

> "Se entiende por expediente electrónico, la serie ordenada de documentos públicos registrados por vía informática, tendientes a la formación de la voluntad administrativa en un asunto determinado." (Artículo 2 Decreto 65/998)
>
> "El expediente electrónico tendrá la misma validez jurídica y probatoria que el expediente tradicional. La documentación emergente de la trasmisión a distancia, por medios electrónicos, entre dependencias oficiales, constituirá, de por sí, documentación auténtica y hará plena fe a todos sus efectos en cuanto a la existencia del original trasmitido (Artículo 129 de la Ley Nº 16.002 de fecha 25 de noviembre de 1988)." (Artículo 3 Decreto 65/998)

Firma Electrónica:

> "Se entiende por firma electrónica, el resultado de obtener por medio de mecanismos o dispositivos un patrón que se asocie biunívocamente a un individuo y a su voluntad de firmar." (Artículo 18 Decreto 65/998)

Firma Digital:

> "Firma Digital es el resultado de aplicar a un documento un procedimiento matemático que requiere información de exclusivo conocimiento del firmante, encontrándose ésta bajo su absoluto control. La firma digital debe ser susceptible de verificación por terceras partes, de manera tal que dicha verificación permita, simultáneamente, identificar al firmante y detectar cualquier alteración del documento digital posterior a su firma." (Artículo 2 inciso a) Decreto 382/003)

El objetivo de estos decretos, es lograr una reglamentación para la incorporación de los medios informáticos y telemáticos a la gestión administrativa, adecuándose a las realidades de cada organismo. Los principios fundamentales de los decretos son lograr una mayor eficiencia y eficacia en la actuación administrativa.

2.3 Estudios de aplicaciones existentes

Desde siempre las grandes oficinas u organizaciones han reunido los documentos asociados a un cierto trámite o acto en legajos que han rotulado con un identificador único, creando así lo que conocemos hoy en día como el expediente tradicional. El mismo constituyó una forma organizada de manejo documental para una era tecnológica que no ofrecía alternativas, basando el éxito de su funcionamiento en las oficinas cuyo único objetivo consistía en poder localizar, en cualquier momento, un

determinado expediente. Los problemas de lentitud, extravíos, robos, deterioros, folios faltantes, imposibilidad de acceso, dificultades de almacenamiento y otros recién fueron encontrando alguna solución en los años 80' cuando se difundieron los primeros programas para seguimiento de expedientes, cuyo cometido esencial era el de llevar registro de la ubicación del expediente físico en todo momento. Pero fue recién hacia finales de los años 90´ cuando la baja en los costos de los sistemas de computación, así como la difusión de estándares universales y abiertos (particularmente los de Internet), permitieron el desarrollo de sistemas de Expediente Electrónico en el sentido más amplio del término.

Software de Gestión Documental son todos aquellos sistemas creados para la gestión de grandes cantidades de documentos. Suelen rastrear y almacenar documentos electrónicos o imágenes de documentos en papel. Normalmente, los documentos no tienen una organización clara de sus contenidos, generalmente se encuentran comprimidos y además de texto pueden contener cualquier otro tipo de documentos multimedia como imágenes o videos [11]. Esta definición no tiene en cuenta los trámites estructurados y definidos en el decreto 500/991 [2] sino solo los documentos utilizados ampliamente en sistemas electrónicos.

La gestión de documentos electrónicos es compleja y exige la correcta aplicación de una gran variedad de funciones. Es obvio que un sistema de gestión de documentos electrónicos de archivo (SGDEA) [7] que colme tales necesidades precisa software especializado, que puede consistir en: un módulo especializado, en varios módulos integrados, en software desarrollado a la medida del usuario o en una combinación de varios tipos de programas informáticos. En todos los casos, siempre tendrán que existir procedimientos y políticas que complementen la gestión de forma manual.

Lo que sigue es un detalle de algunas aplicaciones existentes, en primer lugar a nivel nacional y luego a nivel internacional.

2.3.1. Gestión Documental en Uruguay

En el ámbito nacional, dentro de la Administración pública o privada, existen distintas realidades en lo que tiene que ver a la Gestión Documental. Básicamente se pueden diferenciar tres:

1. Organismos que no cuentan con una herramienta corporativa (se realiza un registro manual en diversos libros o no se dispone de una base centralizada de información, sino de múltiples repositorios de información dentro del mismo organismo).
2. Organismos que disponen de un seguimiento de trámites corporativo donde se registran los documentos básicos y los distintos pasos involucrados en su manejo durante el flujo de trabajo.
3. Organismos que poseen un sistema de gestión de documentos electrónicos. Estos sistemas en general permiten gestionar tanto trámites en soporte electrónico como soporte papel o incluso mixto.

El objetivo principal de la Gestión Documental a nivel nacional basado en la legislación, es la implementación de un producto de gestión de expedientes que posea una capacidad de parametrización que permita adaptarse a la realidad de cada organismo, debe ser multiplataforma informática, de forma de ser compatible con los requerimientos de cada organismo y cumplir con los estándares definidos por "Grupos de Intercambio de Información y Firma Electrónica". En resumen, debe contribuir a la transparencia y transformación del Estado a través de la implantación extensa e interconexión de sistemas de expediente electrónico.

A continuación se mencionan algunos sistemas de Gestión Documental implementados en Uruguay, mencionando algunas de sus características.

EXPE+ [13]

Este es un Sistema de Seguimiento de Expedientes de la Universidad de la República, que se implantó en el 2000. Aquí se define a un Expediente Electrónico como una serie ordenada de documentos, que se tramitan por vía informática, que contiene rutas del proceso, reglas de decisión, contenido de información y usuarios válidos, y que conceptualmente fue generado para el cumplimiento de determinado fin.

Este sistema fue desarrollado a partir del sistema que la Intendencia Municipal de Montevideo (IMM) utiliza para su gestión interna (SEM: Sistema de Expedientes Municipales). Este permite el acceso a información del trámite de cualquier expediente desde cualquier oficina de la IMM y fue adaptado e implantado en la Universidad. Con el correr del tiempo se realizaron diferentes implementaciones, las que se detallan a continuación.

El Sistema de Seguimiento de Expedientes (EXPE+)

Registra los movimientos de notas y expedientes en papel dentro de la Universidad manteniendo la información de actuantes, dependencias involucradas, plazos, etc. Permite la búsqueda de documentos por número, nombre, cédula de identidad, fecha y por alguna palabra o característica que identifique el objetivo de la búsqueda o consulta. También se puede detectar la etapa del trámite en que se encuentra un expediente y los pasos que ha seguido. Las Facultades, que tienen incorporado el expediente electrónico, y los funcionarios pertenecientes a ellas, podrán obtener los formularios de inicio de los mismos.

Sus características generales más importantes son:

- Permite el ingreso y la consulta de información relativa a notas y expedientes desde cualquier dependencia de la institución que tenga instalado el software.
- Ofrece la posibilidad de obtener información precisa y actual acerca de la ubicación física de los documentos que hayan sido iniciados por cualquiera de los puestos de trabajo en toda la Universidad.
- Asigna un número único en toda la Universidad a cada nota o expediente.

- Dispone de un rápido y amplio sistema de búsqueda permitiendo acceder a un documento conociendo alguno de sus datos (número; cualquier palabra o frase; nombre, cédula o dirección del interesado; dependencia de inicio, ubicación actual, funcionario interviniente, etc.).
- Controla vencimientos de plazos establecidos para la tramitación de documentos (detectar los documentos estancados más tiempo del previsto en función del motivo de pase en cada punto del recorrido, los documentos que superan los plazos totales previstos para su tramitación y controlar vencimientos definidos en cada dependencia).
- Brinda datos estadísticos acerca de la generación y tramitación de documentos en toda la Universidad, permitiendo contar con información confiable para la toma de decisiones. Automáticamente permite saber para cada dependencia en un período determinado, la cantidad de documentos iniciados, recibidos y enviados. Los usuarios pueden crear sus propios perfiles de estadísticas, almacenarlos y solicitar que se calculen cuando se lo necesiten.
- Permite la consulta vía navegador web: www.expe.edu.uy.
- Cuenta con un "Manual del Usuario" en línea que permite la consulta por parte de usuarios que utilizan el sistema.
- Cuenta con una "Base de Discusión" en línea que permite realizar sugerencias y consultas hacia el proyecto y su devolución, compartible con todos los usuarios.

El Sistema de Expediente Electrónico

Maneja documentos electrónicos sin soporte en papel, las actuaciones se realizan directamente en el formulario electrónico y se firman digitalmente quedando registrados además los movimientos en un sistema de seguimiento. Se basa en un sistema de flujos de trabajo donde cada actividad firmada digitalmente determina destino y actuantes de la siguiente actividad quedando definido así un proceso de ruta fija.

El Sistema de Gestión de Resoluciones (SGR)

La interacción con las necesidades de la gestión permitieron la planificación, desarrollo e instalación de este sistema que no estaba en el marco del proyecto inicial a partir del cual se creó el sistema de expedientes. Es un software de e-government que organiza y automatiza todo el proceso de las sesiones de Consejos y Comisiones Directivas de la UdelaR. El mismo fue desarrollado enteramente por el equipo de informáticos de la Universidad.

El sistema de Resoluciones permite registrar todo el proceso relacionado a la generación y publicación de las resoluciones tomadas en una sesión, por un órgano decisor (Consejo Directivo Central, Consejo Ejecutivo Delegado, Consejo de una Facultad y/o Comisión Directiva), y ofrece la posibilidad de recuperar las resoluciones adoptadas en sesiones anteriores, así como toda otra documentación utilizada en las mismas.

Es posible realizar todas las actividades relacionadas al proceso, desde el ingreso del proyecto, hasta la publicación de la resolución en la página web. Además tiene una potente herramienta de búsqueda que permite ubicar documentos por cualquier dato incluido en los mismos.

Es utilizado por los funcionarios encargados de preparar la documentación necesaria para la sesión del órgano decisor, a los que se les asigna distintos roles, de acuerdo a la función que cumplan.

Permite obtener indicadores estadísticos más fácilmente.

Las Resoluciones constituyen una de las actuaciones más importantes de las tramitaciones.

Características del Sistema SGR:

- Actividades automáticas a demanda (ej.: generación del orden del día a partir de proyectos de resolución, generación de repartidos, etc.).
- Trabajo en paralelo (varias personas a la vez) en el ingreso de proyectos de resolución y modificación de los textos aprobados.
- Utilización de plantillas para los textos de resolución.
- Fácil acceso a la información. Posibilidad de consulta por cualquier palabra o frase de los proyectos de resolución, resoluciones postergadas, resolución aprobada, orden del día, repartido, etc.
- Posibilidad de envío automático de mail con el orden del día, alcances, actas y repartidos.
- Publicación automática en el Web de las resoluciones aprobadas. Consulta de ellas por cualquier palabra o dato contenido en la misma.
- Respaldo informático automático en diversos dispositivos (disco de un servidor, cintas, CD's).

GEX [14]

GEX es un sistema de gestión de expedientes electrónicos implementada por UTE e implantada en varias dependencias públicas.

Los principales beneficios de este sistema son:

- Mejoras en la gestión mediante la incorporación de documentos electrónicos, organización en los procedimientos administrativos, búsqueda efectiva de expedientes y documentos, valiosos reportes de volúmenes que permiten un mejor manejo de la información requerida para una buena gestión.
- Transparencia de la gestión de las distintas formas documentales implementando los mecanismos necesarios para saber en cada momento donde se encuentra cada forma documental y por donde pasó, teniendo así un seguimiento global a nivel del estado.
- Asegurar la accesibilidad a través del tiempo, al no utilizar formatos propietarios para almacenar la información.
- Mejoras en la seguridad y control mediante los mecanismos de firma digital que aseguran la autenticidad e integridad de la documentación.

- Mejoras en la imagen del organismo mediante la mejora de rapidez en los trámites y accesos a la información por parte de los clientes del organismo.

GEX en la Web

Su última versión fue implantada en el 2002.

Abarca varios aspectos sobre el tratamiento de los expedientes y cubre el marco normativo vigente. Algunos de los organismos que lo utilizaron en un principio son: Oficina Nacional de Servicio Civil (ONSC), Ministerio de Vivienda Ordenamiento Territorial y Medio Ambiente (MVOTMA), Presidencia de la República, Banco de la República Oriental del Uruguay (BROU), la Dirección Nacional de Aduanas (DNA), Ministerio de Defensa (MDN) y ANCAP.

GDCWEB

Es la solución de expediente electrónico implantada en UTE con una cobertura funcional básica muy similar a GEX con algunos módulos específicos (como la gestión de Resoluciones) y con un importante software de base (gestor documental, flujo de trabajo, buscador, etc.) adicional.

Desde noviembre de 2004, con la utilización de GEX, se encuentran interconectados todas las Direcciones Generales de Secretaría, los Departamentos de Acuerdos de los Ministerios con Presidencia y ONSC (Oficina Nacional de Servicio Civil), pudiendo realizar envíos electrónicos de expedientes entre ellos.

iGDoc [5]

iGDoc es una solución nacional desarrollada por la empresa ISA Ltda. Es una plataforma que brinda servicios para la Gestión Electrónica de Documentos a diversas formas documentales como: Expedientes, Formularios, Resoluciones, Comunicaciones. Estas Formas Documentales tienen en común requerir de un repositorio de documentos, numeración centralizada, niveles de seguridad para garantizar su acceso y confidencialidad, posibilidad de ser firmados digitalmente, localizarlos mediante un buscador de texto completo y de contenidos, y además ser accesibles desde un navegador Web.

En particular iGDoc incorpora las formas establecidas por el marco normativo vigente, tanto aquellas definidas para documentar y dar a conocer la voluntad administrativa en los procedimientos que se sustancien por escrito como las que sirvan a las comunicaciones escritas entre distintas reparticiones

Las formas incorporadas a esta infraestructura son: iGDoc expedientes (su anterior versión es File Center), iGDoc formularios, iGDoc resoluciones, iGDoc comunicaciones, iGDoc biblioteca.

Constituye una plataforma extensible que permite implementar con mínimos costos otras formas documentales específicas para una determinada organización, sobre la misma infraestructura y utilizando las mismas capacidades probadas del repositorio.

Desde el punto de vista de la interface de usuario requiere solamente de un navegador Web. Se vincula con herramientas para el trabajo colaborativo, la comunicación empresarial y la integración de aplicaciones corporativas para ofrecer un escritorio donde se pueda visualizar el repositorio de documentos integrado con servicios de correo electrónico, mensajería instantánea y otras aplicaciones.

Posee una arquitectura abierta basada en estándares para el intercambio de documentos, la exposición de servicios y la firma digital, puede inter-operar con otros sistemas transaccionales o documentales de la misma u otras organizaciones.

Es una solución multiplataforma lo que significa que puede ser instalada en una amplia variedad de equipos y sistemas operativos.

Q-expeditive [4]

Este es un sistema desarrollado por la empresa Urudata. Es un sistema de expediente electrónico, o dicho de otra forma, una herramienta que permite definir gráficamente los trámites internos que una empresa lleva a cabo, modelarlos y ejecutarlos ordenadamente, integrando la información de los sistemas administrativo-contables, y proporcionando así una mejora en la eficiencia de los mismos.

Provee la posibilidad de digitalización de documentos físicos existentes para que el manejo de legajos se lleve a cabo en forma 100% digital. Automatiza los pasos de los trámites, también llamados actuaciones, implementando la firma digital mediante el uso de certificados, y la seguridad de acceso requerida en toda gran organización.

Beneficios de usar Q-expeditive: mayor eficiencia operativa, prevención de atrasos, disponibilidad de información del estado de un trámite para consulta del público, defensa contra alteraciones malintencionadas de documentos, registro automático de información para auditoría.

Ventajas de Q-expeditive: reducción de pérdidas de información por deterioro, extravío y robos, interoperabilidad con sistemas externos, posibilidad de representar fácilmente trámites complejos con rutas paralelas y caminos alternativos, inicio de trámites a través de Internet.

DoMUS [84]

DoMUS (Document Managment) es una herramienta desarrollada por la empresa Arnaldo C. Castro S.A. Se integra con scanners de producción,

permitiendo el trabajo concurrente, optimizando la carga e indexación masiva de documentos.

En los entornos organizacionales actuales, se maneja cada vez más información, siendo un alto porcentaje de la misma no estructurada (documentos electrónicos, imágenes, cartas, faxes, tarjetas, libros, expedientes). Ésta no puede ser almacenada dentro de una base de datos tradicional, por lo que la dificultad de su gestión crece de forma exponencial en función de su volumen. Esta realidad exige cada vez más una herramienta que ayude a gestionar la información de forma eficiente y segura, permitiendo la integración con los procesos funcionales y operacionales de la empresa tanto en el ámbito individual como departamental o corporativo. DoMUS es la propuesta para realizar esta gestión.

DoMUS fue concebido basándose en los avances tecnológicos en materia de medios de almacenamiento y dispositivos de captura aplicados al desarrollo de sistemas de Document Imaging, brindan muchas ventajas con respecto a la manipulación manual de la información, ya que estos sistemas permiten la captura, almacenamiento, distribución, visualización, impresión, clasificación y seguridad de cualquier documentación que esté en un soporte de papel o soporte digital archivos de procesadores de texto, imágenes, fotografías, etc.

Es un sistema orientado a Internet y está desarrollado con JAVA utilizando estándares como XML, JSP, ISIS, Firma Digital, etc. El servidor es multiplataforma y puede ser instalado sobre diferentes sistemas operativos (Windows NT, 2000, XP, 2003, Solaris, AIX, Linux) y puede comunicarse con Bases de Datos: Oracle, Microsoft SQL Server, DB2, Informix, MySQL y PostgreSQL.

INTEGRADOC [6]

Es un sistema desarrollado por la empresa Integradoc, de Gestión Documental y Expediente Electrónico completo, con flujos ad-hoc o predefinidos, que permite integrar nuevos procesos y tipos de documentos, y comunicarse con otros sistemas y organizaciones mediante la utilización de estándares. Ofrece el proceso completo de digitalización de expedientes en papel, incorporando la herramienta DoMUS mencionada anteriormente.

Permite manejar diversos tipos de documentos, con sus procesos de negocio asociados, proveyendo un repositorio único de los mismos, que genera automáticamente indicadores de rendimiento sobre el funcionamiento de los procesos, imprescindibles para la toma de decisiones.

Basado en estándares abiertos de la industria, garantiza la integración con otros aplicativos así como con otras instalaciones de Integradoc y de Expedientes Electrónicos. La solución es multiplataforma.

Los organismos de gobierno, generalmente tienen un intenso manejo de diferentes tipos de documentos, que fluyen por el organismo mientras los funcionarios toman determinadas acciones sobre ellos. De esta forma expedientes, notas, circulares, oficios, resoluciones y formularios son generados y enviados de una persona a otra para que realice su trabajo y el mismo quede debidamente documentado. Este sistema permite la sustitución del soporte en papel para todos estos tipos de documentos, permitiendo definir flujos ad-hoc así como predefinidos, para cada tipo de documento y para cada tema. Integradoc además almacena los documentos en un repositorio centralizado facilitando su búsqueda y recuperación. La seguridad de estos documentos es máxima, contando con firmas digitales y respetando la normativa legal vigente. Genera automáticamente indicadores de los procesos, tales como cantidades de documentos creados y procesados, pudiendo navegar sobre estos informes por tipo de documento, tema, fecha de creación entre otros atributos.

Las organizaciones para-gubernamentales usualmente tienen necesidades similares pero no idénticas a los organismos gubernamentales respecto a la gestión documental involucrada en su funcionamiento. Es así que además de los tipos de documentos típicos de las organizaciones gubernamentales, tienen necesidades específicas como ser el manejo de minutas de reuniones, actas de directorio, resoluciones de directorio, memorandos, etc. Este sistema facilita definir los flujos para cada uno de estos tipos de documentos, prestando un soporte completo a los mismos, manejando las asignaciones, las alertas, los plazos de vencimiento y generando estadísticas automáticas que presenten datos objetivos y medibles referentes a como funcionó cada proceso. Cuenta con altos niveles de seguridad y auditoría, lo que lo convierte en una herramienta para automatizar, posteriormente gestionar y finalmente mejorar los procesos de las organizaciones para-gubernamentales.

Se basa en una plataforma diseñada para manejar grandes volúmenes de datos, como son las bases de datos relacionales, en una arquitectura J2EE.

Resumen

A continuación el detalle de los principales sistemas de seguimiento o gestión de expedientes electrónicos que se encuentran implantados en Uruguay, junto con el detalle de las tecnologías utilizadas:

- ANP (Administración Nacional de Puertos): Sistema iGDoc.
- BROU (Banco de la República Oriental del Uruguay): Sistema GEX en la web, octubre de 2003. Basado en Visual Basic / ASP/ SQL Server.
- CGN (Contaduría General de la Nación): Sistema Lotus Notes versión 5.0.10, proveedor Datamatic y desarrollo de parametrización propio

sobre Lotus Domino. El sistema no permite gestionar los expedientes y se limita a un seguimiento.

- DNA (Dirección Nacional de Aduana): Sistema GEX en la web, octubre de 2004. Basado en Visual Basic / ASP/ SQL Server.
- MDN (Ministerio Nacional de Defensa): Sistema GEX en la web.
- MEC (Ministerio de Educación y Cultura): Sistema iGDoc.
- MEF (Ministerio de Economía y Finanzas): Desarrollo tercerizado por la CGN sobre Lotus Notes versión 5.0.10. Proveedor Datamatic con desarrollo propio sobre Lotus Domino. Se destaca la posibilidad de adjuntar documentos en varios formatos como Microsoft Word y Acrobat.
- MGAP (Ministerio de Ganadería Agricultura y Pesca): Sistema SIADOC (sistema informático de administración documental), sobre Visual Basic / web con SQL Server.
- MI (Ministerio del Interior): Sistema propio desarrollado por el Departamento de informática sobre Genexus, en Visual Basic y con SQL Server. El sistema no permite gestionar los expedientes y se limita a un seguimiento.
- MIDES (Ministerio de Desarrollo Social): Sistema Q-Expeditive.
- MSP (Ministerio de Salud Pública): Lotus Notes. Clientes 5.012 Versión Servidor. El sistema no permite gestionar los expedientes y se limita a un seguimiento.
- MTOP (Ministerio de Transporte y Obras Públicas): Sistema basado en Lotus Notes, Versión Cliente 5.02 Versión Servidor 5.3. El sistema no permite gestionar los expedientes y se limita a un seguimiento.
- MTSS (Ministerio de Trabajo y Seguridad Social): Sistema propio denominado Administración Documental, desarrollado de manera mercerizada. El programa permite hacer un seguimiento de los expedientes.
- MVOTMA (Ministerio de Vivienda, Ordenamiento Territorial y Medio Ambiente): Sistema GEX EN LA WEB, octubre de 2003. Basado en Visual Basic / ASP/ SQL Server.
- ONSC (Oficina Nacional de Servicio Civil): Sistema GEX EN LA WEB, octubre de 2003. Basado en Visual Basic / ASP/ SQL Server.
- PRESIDENCIA DE LA REPÚBLICA: Sistema GEX EN LA WEB, implantado en enero de 2001.
- Intendencia Municipal de Maldonado: Sistema iGDoc.
- Intendencia Municipal de Canelones: Sistema iGDoc.
- Intendencia Municipal de Florida: Sistema iGDoc, siendo ésta la primer Intendencia del Uruguay en contar con un sistema de Expediente Electrónico para el 100% de sus oficinas.

2.3.2. Gestión Documental en otros países

A nivel internacional las empresas también realizan una gestión documental y manejo de flujos de trabajo de diversas formas. A continuación se detallan algunos sistemas interesantes a los cuales se tuvo acceso.

Especificación MoReq [7]

En la comunidad europea se define el manejo de trámites mediante un modelo de requisitos para la gestión de documentos electrónicos de archivo: la especificación MoReq.

Es un modelo de requisitos funcionales para la gestión de documentos electrónicos de archivo elaborado a través del Programa IDA (Intercambio de Datos entre Administraciones) con el fin de que pueda ser utilizado en todos los países de la Unión Europea y por todos los interesados en el desarrollo y aplicación de sistemas de gestión de documentos electrónicos de archivo (archiveros, gestores, diseñadores de software, proveedores de servicios, instituciones académicas y de formación).

GEDEX [8]

Es un software jurídico para gestión de expedientes utilizado en España e Hispanoamérica, para abogados, procuradores y profesionales, disponible en castellano, catalán e inglés, permitiendo trabajar en despachos de un único letrado, bufetes y departamentos jurídicos de empresa, tanto en computadores individuales como en redes locales.

Es un producto adaptable a muy distintos modos de gestión, integrándose a un despacho sin cambiar el tratamiento de la información al que se encuentre habituado. Es por ello que abogados, peritos, procuradores, organismos oficiales españoles, empresas privadas, asesorías y facultades de Derecho lo utilizan profesionalmente.

Clasifica expedientes instantáneamente en función de múltiples aspectos (plazos, estado de apertura, etc.).

Ofrece un avanzado sistema de contraseñas, con el que puede limitar el acceso a la información, ocultar expedientes y contactos a ciertos pasantes o empleados, bloquear los cambios sin autorización, entre otras funcionalidades. Permite integrarse con la configuración de seguridad de su red local, gestionar su propio esquema de seguridad o asignar contraseñas específicas, según sea su despacho.

sdmEXP – Gestión de expedientes administrativos [9]

SdmEXP es una aplicación realizada por una empresa española, que permite realizar un seguimiento completo de cada uno de los expedientes administrativos que se gestionen en un despacho profesional, con posibilidad de crear modelos de expedientes que le permitan saber los tramites, impresos, formularios, o incluso los movimientos económicos que se deben realizar para llevar a cabo cada uno de los expedientes que entran en un despacho profesional.

Incluye cerca de 500 impresos oficiales, tanto estatales como autonómicos, así como todos los de la AEAT (Agencia Tributaria de España) [10], que se

van incrementando automáticamente desde la propia aplicación. También le permite crear modelos de formularios para utilizar en los distintos expedientes que vaya a gestionar.

Está integrado con la aplicación de Facturación de Despachos Profesionales lo que le permite realizar automáticamente las facturas de los expedientes correspondientes.

Gestiona fácilmente los expedientes administrativos de un despacho profesional.

VERS [85]

"The Victorian Electronic Records Strategy" (VERS) ha sido desarrollado por Public Record Office Victoria para conservar los registros electrónicos del Estado a largo plazo. Es un estándar para la definición de expedientes que fue diseñado para asistir a la administración pública en el manejo de expedientes electrónicos en el Estado de Victoria de Australia. La estrategia se enfoca en los datos o información contenida en los mismos, en vez de los sistemas usados para producirlos. Estos estándares cubren aspectos como la creación, mantenimiento, administración, destrucción y transferencia de los expedientes públicos.

La estrategia de VERS provee una plataforma sobre la cual es posible capturar y archivar expedientes electrónicos, en un formato independiente de un sistema en particular (hardware o software). El modelo VERS considera que los expedientes se pueden almacenar en carpetas, llamando a este concepto "archivos" (file). Soporta además la agregación de datos (información) relacionada con un tema en particular, en donde propone su manejo al nivel de archivo en vez de al nivel de expedientes.

El enfoque se basa en el uso de estándares reconocidos en el área de software y almacenamiento (como por ejemplo XML), en vez del uso de aplicaciones particulares que pueden cambiar con el paso del tiempo y volverse incompatibles con los requerimientos de la preservación de los expedientes.

El enfoque para la preservación de los expedientes requiere que sea algo a largo plazo, pero la realidad es que los sistemas informáticos y aplicaciones cambian o se vuelven obsoletos muy rápidamente. Muchos problemas han sido identificados como impedimentos para el manejo a largo plazo de los expedientes electrónicos.

- El formato de los documentos cambian y se vuelven obsoletos en el tiempo.
- Los objetos electrónicos están propensos a cambios indetectables, haciendo más difícil de mantener el estado de evidencia y responsabilidad sobre los expedientes.
- El contexto de un expediente electrónico y su relación con otros expedientes puede fácilmente perderse.

- Los sistemas existentes para el manejo de documentos electrónicos no preservan la integridad del contenido, estructura y contexto de éstos durante el tiempo que el expediente es requerido.

Cada uno de estos problemas ha sido tomado en cuenta y solucionado en el desarrollo del VERS.

Capitulo 3 Análisis de Requerimientos y Tecnologías

3.1. Introducción

En este capítulo se detalla el análisis realizado sobre los requerimientos del sistema y el de las tecnologías investigadas para satisfacerlos.

3.2. Requerimientos

Cómo se explicó en los capítulos anteriores, algunos de los distintos trámites están definidos en el Decreto 500/991 [2] de la actual legislación uruguaya, entre ellos se encuentran las formas documentales, expedientes, formularios, resoluciones y comunicaciones, que son las contempladas en esta sección de requerimientos.

Tanto la legislación como las necesidades de las distintas organizaciones pueden cambiar a lo largo del tiempo, se pueden definir nuevas formas documentales o realizar modificaciones en las ya existentes. Por lo tanto se requiere que el proyecto considere estos posibles y muy probables cambios en el negocio, para que el mismo sea extensible y flexible, y que permita con el menor esfuerzo posible, adaptarse a la nueva realidad.

Por esta razón en el sistema se van a definir diferentes capas de abstracción, una capa que ofrece servicios básicos para la gestión de documentos que pueda ser utilizada por todas las formas documentales y luego sobre esta capa se podrán desarrollar los diferentes módulos funcionales que implementan las particularidades de cada Forma Documental. De esta manera se facilita la creación de otros módulos que representen e implementen futuras formas documentales.

También se requiere contemplar particularidades de cada organización, como por ejemplo, que la organización ya disponga de un sistema que implemente una forma documental, por lo tanto cada modulo funcional debe poder funcionar independiente de los demás módulos funcionales.

Los requerimientos del Sistema de Gestión Documental se subdividen en dos grandes módulos: los módulos generales y los módulos funcionales del sistema. Los módulos generales van a ser usados por los módulos funcionales. Por otra parte también se encuentran los requerimientos no funcionales generales que debe cumplir toda la solución.

A continuación se describen brevemente estos módulos y se incluye un anexo [Anexo A] donde se encuentra el detalle de los requerimientos.

3.2.1. Módulos Generales

En esta sección se nombran y describe brevemente cada uno de los módulos generales:

- Gestión de Documentos
  Ofrece las funcionalidades generales para todas las formas documentales.
- Numerador
  Permite crear numeradores que permitan numerar consecutivamente los diferentes documentos.
- Seguridad
  Se configuran los criterios de seguridad para todas las formas documentales que estén disponibles en la organización.
- Administración de Usuario
  Se mantiene la gestión de usuarios de la organización de forma genérica para todos los módulos funcionales de la aplicación.
- Preferencias de Usuarios
  Configura las preferencias de los usuarios.
- Estructura de la Organización
  Ofrece el mantenimiento de las unidades y su jerarquía dentro de la organización.
- Auditoría
  Registro y consultas de las actividades de los usuarios en el sistema.
- Registro de Errores
  Se registran todos los errores ocurridos en la aplicación para ofrecer un menor soporte y facilitar el mantenimiento de la misma.
- Calendario
  Registros de días no hábiles para las estadísticas, reportes y notificaciones.
- Estadísticas y Reportes
  Permite obtener reportes y estadísticas para mejorar los procesos de la organización.

3.2.2. Módulos Funcionales

A continuación se nombran y se describen brevemente cada uno de los módulos funcionales:

- Expedientes
  Se encarga del manejo de la forma documental expedientes.

- Resoluciones
  Es el encargado de la gestión de las resoluciones dentro de la organización.

- Formularios
  Maneja la gestión de la forma documental formularios.

- Comunicaciones
  Ofrece las funcionalidades para las comunicaciones entre las unidades de la organización.

- Gestión de Bandejas
  Realiza el mantenimiento de bandejas de las unidades y usuarios para el acceso de las diferentes formas documentales.

3.2.3. Requerimientos no funcionales

A continuación se nombran los requerimientos no funcionales más importantes:

- Tecnológicos
  - Tecnologías open-source
  - Lenguaje Java
  - Representación de XML
  - BPM
  - Búsqueda full-text

- Mantenimiento
  - Escalable
  - Independencia de los módulos

- Portabilidad

- Seguridad

3.3. Tecnologías

En esta sección se resume la investigación de diferentes tecnologías que puedan ser utilizadas para satisfacer los requerimientos del sistema. El mismo se dividirá en secciones enfocadas a los principales requerimientos que definen la arquitectura de la solución.

En primer lugar se analiza la definición y la ejecución de los distintos workflows que va a tener la aplicación. Para la definición se buscará un editor gráfico que ayude a satisfacer el requerimiento de diseñar el proceso que determina las unidades por las que pasarán las distintas formas documentales, expedientes, formularios, resoluciones, etc. Al modelo de este proceso se lo denominará ruta de unidades. La ejecución debe

contemplar la ruta entre oficinas y el manejo de estados de los documentos. Para ambos casos se estudia el uso de estándares de definición de procesos, como puede ser XPDL.

En segundo lugar se analizan los distintos tipos de bases de datos, evaluando cual es la mejor opción para la problemática planteada, teniendo en cuenta la utilización de XML para las formas documentales.

En tercer lugar, como un anexo posible a utilizar junto con la base de datos, se investigarán distintos repositorios de archivos, para manejar los archivos adjuntos de las formas documentales y permitir realizar búsquedas de información.

En cuarto lugar, se analizan los distintos requisitos que debe contemplar una herramienta web para poder ser tenida en cuenta al momento de realizar el prototipo del sistema.

Para trabajar con los archivos XMLs se estudiaron diferentes herramientas que facilitan su manejo.

Para más información se incluye un anexo [Anexo B – B.2] con especificaciones técnicas de las herramientas investigadas en mayor profundidad.

3.3.1. Definición de rutas y workflow de los documentos

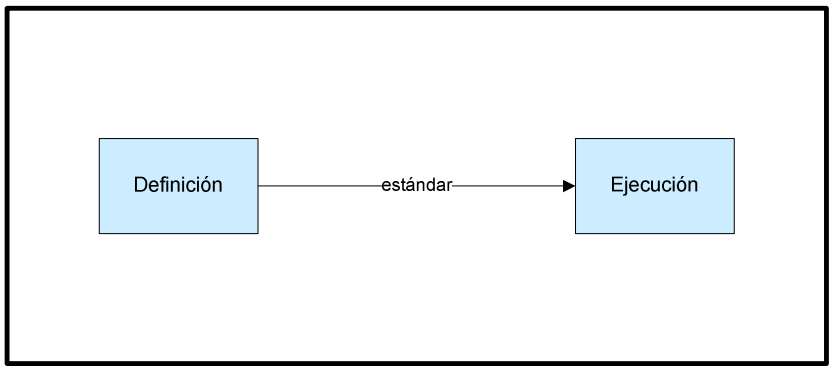
Dentro de los principales requerimientos del proyecto, se encuentra la *definición* y *ejecución* de rutas de unidades por las que pasarán las diferentes formas documentales. Las mismas pueden ser rutas fijas, libres o híbridos de las anteriores.

Las rutas son *definidas* por un usuario administrador de forma gráfica e integrada con la aplicación, es decir, podrá crear rutas especificando para cada una de las unidades seleccionadas cuál es la próxima encargada de procesar la forma documental dependiendo de determinadas propiedades de esta última.

La *ejecución* utiliza esta definición para realizar los pases entre las unidades.

La comunicación entre la *definición* y la *ejecución* se realiza mediante una estructura determinada, para lo cual se puede utilizar un estándar de definición de procesos, como por ejemplo XPDL o BPEL (ver Anexo B.1).

A estas partes se las considera como dos cajas negras, donde la *definición* tiene como salida la estructura y la *ejecución* la recibe y la procesa, como se muestra en la figura 2.



2- Utilización de un estándar

Existen otros requerimientos de workflows para los cambios de estados de los documentos y el manejo de los mismos dentro de una unidad. En este caso los workflows son más estables y no requieren ser definidos por los usuarios.

A continuación se describen las alternativas estudiadas para la *definición* y *ejecución*, analizando ventajas, desventajas y herramientas existentes para cada una. En la ejecución se analizan las alternativas para los diferentes requerimientos presentados anteriormente (ruta de unidades, cambios de estados de los documentos y proceso del documento dentro de una unidad). En un anexo [Anexo B – B.1] se incluye información más profunda sobre flujos de trabajo, las metodologías de negocio y estándares sobre los que se apoyan.

3.3.2. Definición de rutas de unidades – Editores gráficos

Para la definición gráfica de las rutas de unidades se plantean dos alternativas: la posibilidad de utilizar un editor de workflow externo que genere un archivo que respete un estándar para luego ser ejecutado por un motor de workflow que interprete el estándar, o que la propia aplicación ofrezca embebido el editor.

En esta sección se detalla cada alternativa y luego se realiza la comparación de las mismas.

EDITOR EXTERNO

Existen varias herramientas de edición de workflow que permiten el diseño gráfico del mismo, generando una estructura basada en los estándares de definición y/o ejecución de workflows como son BPEL o XPDL (ver Anexo B.1). En el Anexo B.2 se listan algunas de las herramientas existentes y a continuación se detallan las que se investigaron con mayor profundidad.

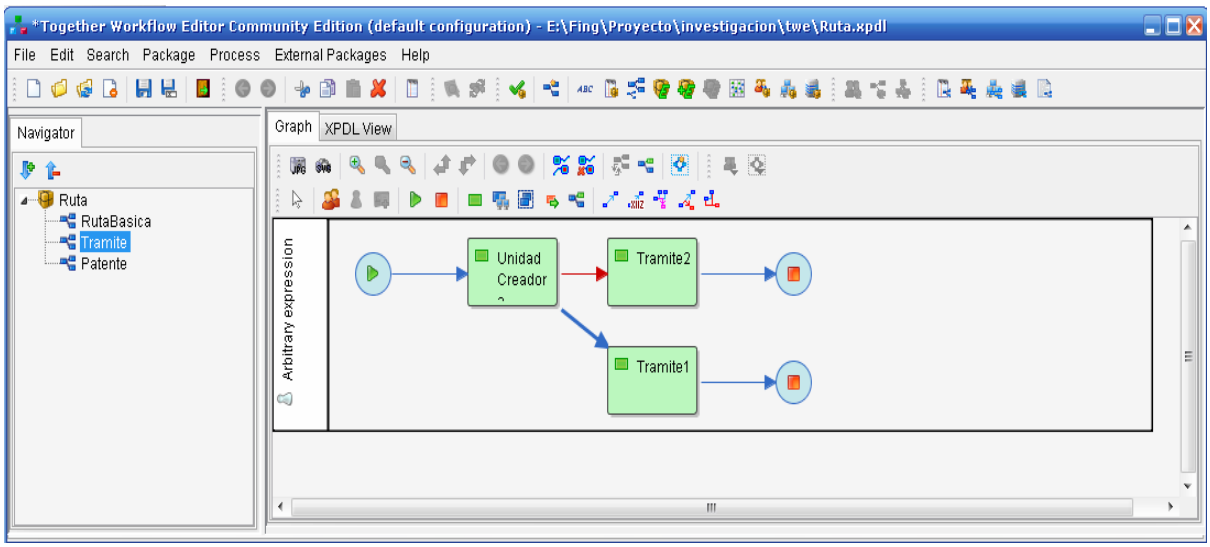
Together Workflow Editor [15]

Este editor permite realizar el diseño gráfico del workflow, generando automáticamente el XPDL que representa el workflow diseñado.

El XPDL contiene la mayoría de las entidades que puedan ser necesarias en el proceso de definición de modelos.

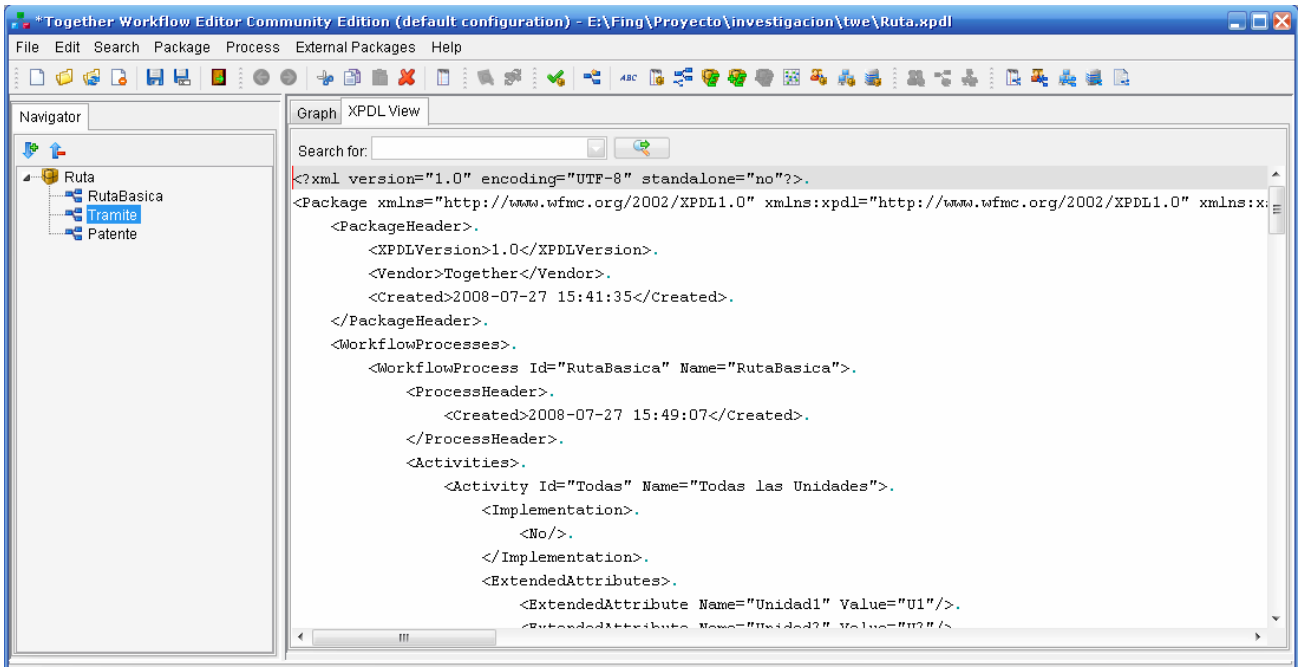
En Together Workflow Editor existe la posibilidad de manejar, al momento de diseñar información adicional del negocio, los extended attributes. Los extended attributes pueden ser usados en todas las entidades (actividades, transacciones, etc.). El usuario puede seleccionar un extended attribute de la lista de los ya creados para el tipo de elemento en el que se encuentra. La lista de nombres se calcula en memoria cada vez que nuevos extended attributes se añaden. Dicha lista no es persistida en ningún lugar.

Para este proyecto se necesitarían utilizar estos extended attributes para definir parámetros especiales de la aplicación. Para que el usuario pueda seleccionarlos de una lista deben estar creados previamente en un elemento del mismo tipo. Para lograr esto la aplicación debería generar un XPDL que contenga un diseño base que defina todos los extended attribute que se necesiten en cada entidad y el usuario debería realizar los diseños de las rutas de unidades utilizando el XPDL creado por la aplicación como una planilla. Esta forma de trabajo limita la posibilidad de restringir las próximas unidades a dar pase, entre otras validaciones. Por lo cual, luego de que el usuario realice el diseño completo se deberían realizar las validaciones de las reglas del negocio, esto es un punto negativo para la usabilidad de la aplicación.



3- Definición de un workflow gráficamente en Together Workflow Editor

En las figuras 3 y 4 se muestra un ejemplo de la definición de una ruta utilizando esta herramienta, en la primera aparece gráficamente y en la segunda el XPDL generado.



4- Definición en XPDL de un workflow en Together Workflow Editor

Se evalúa la alternativa del editor externo describiendo sus ventajas y desventajas como solución a los requerimientos:

Ventajas:

- Genera un archivo que cumple con un estándar, con el cual se podría llegar a cambiar la aplicación externa que se usa, por otra que genere el mismo estándar sin demasiadas modificaciones en la aplicación. Al utilizar estándares se obtiene una solución más extensible.
- Por ser una aplicación externa tiene su propia evolución y su propio testing.
- Se podrían ampliar los requerimientos de modelos a diseñar, utilizando el estándar en su máxima expresión.

Desventajas:

- Usabilidad para el usuario. Ofrece más funcionalidades de las necesarias y para limitar al usuario su utilización se deberá modificar el código fuente.
- Herramienta externa que se debe acceder por afuera de la aplicación.
- Difícil integración con la aplicación para obtener datos.
- Validación del negocio posterior a la finalización del diagrama.

EDITOR INTEGRADO

En esta alternativa, la idea es ofrecer la definición de las rutas de unidades gráficamente de manera más específica para el problema planteado, permitiendo realizar solo las funciones que son necesarias.

Para facilitar la implementación, se investigaron distintas herramientas externas open-source que ofrezcan la interfaz para el diseño del diagrama. Luego, la aplicación interpretaría la salida de esta herramienta para generar el workflow diseñado por el usuario. La herramienta podrá generar un estándar al igual que las herramientas estudiadas para la sección anterior o generar una estructura que sea interpretada por la aplicación.

Entre las herramientas estudiadas se encuentran mxGraph [16] y GEF [17], las que se describen a continuación.

MxGraph [16]

Es una librería JavaScript que utiliza capacidades del navegador para proporcionar una interfaz interactiva de dibujo y diagramación de soluciones. Permite a los clientes dibujar y compartir diagramas. Ofrece:

- Actualización centralizada administrada en el servidor.
- No requiere instalación de plugins en el cliente.
- Se puede hacer deploy utilizando Java, .Net, PHP, HTML estático.
- Fácil configuración usando XML.
- Interfaz de usuario en HTML y datos intercambiados en XML.

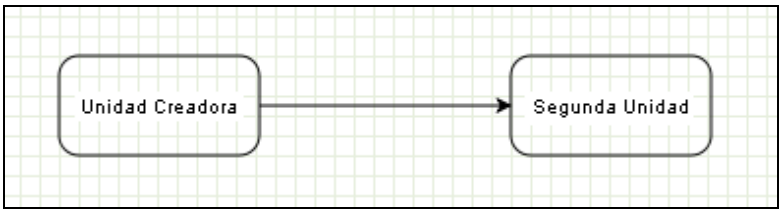
Este no es un software libre, pero se decidió estudiarlo por las facilidades que ofrece y porque es open-source.

Para este proyecto se utilizaría una versión de evaluación.

Este software posee un código que puede ser modificado, y mediante el uso de las funciones JavaScript [18], fácilmente se pueden acotar las opciones dadas al usuario, para restringirlas a lo que la aplicación necesita, como también se pueden agregar funcionalidades. La idea general es integrar el código de mxGraph al de la aplicación pasando la información necesaria para definir la ruta (como mínimo las unidades existentes), luego, desde el código de mxGraph obtener esa información y utilizarla para mostrarla al usuario de forma adecuada.

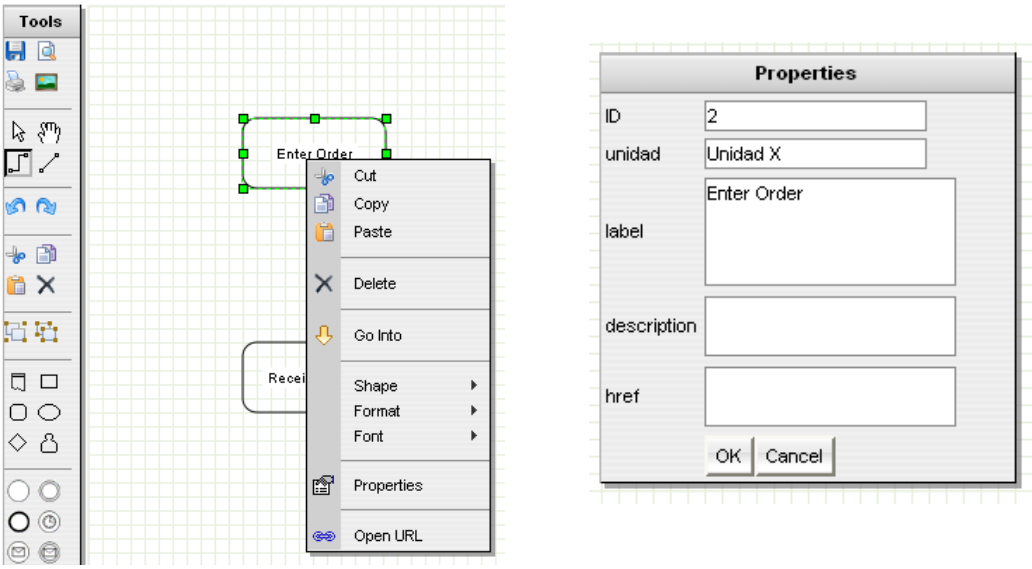
El dibujo puede ser realizado de forma sencilla por parte del usuario, y genera un XML no estándar. La aplicación debería tomar ese XML, transformarlo en XPDL y procesarlo para obtener la ruta definida. La transformación a XPDL es requerida por el cliente, ya que la idea es que los módulos se comuniquen mediante estándares para mantener la posibilidad de que luego distintas herramientas puedan ser cambiadas. Las validaciones de la ruta de unidades en proceso de definición pueden llegar a realizarse mientras se dibuja, esto le da interacción con el usuario.

En la figura 5 se presenta un ejemplo de cómo quedaría una especificación de una ruta de unidades con esta herramienta.



5- Definición de un workflow gráficamente en MxGraph

Al hacer click derecho sobre cada elemento se accede a un menú, y desde él se pueden setear las propiedades del elemento. En el ejemplo que sigue, se muestra el cuadro de dialogo de las propiedades con el código modificado (en parte). Por ejemplo, aquí se eliminaron las propiedades de tamaño que se permiten setear originalmente (de todas formas el tamaño de los elementos se puede seguir modificando con el mouse al arrastrar), y se agrego la propiedad unidad, esto es solo a modo de ejemplo para visualizar la flexibilidad que ofrece esta herramienta.



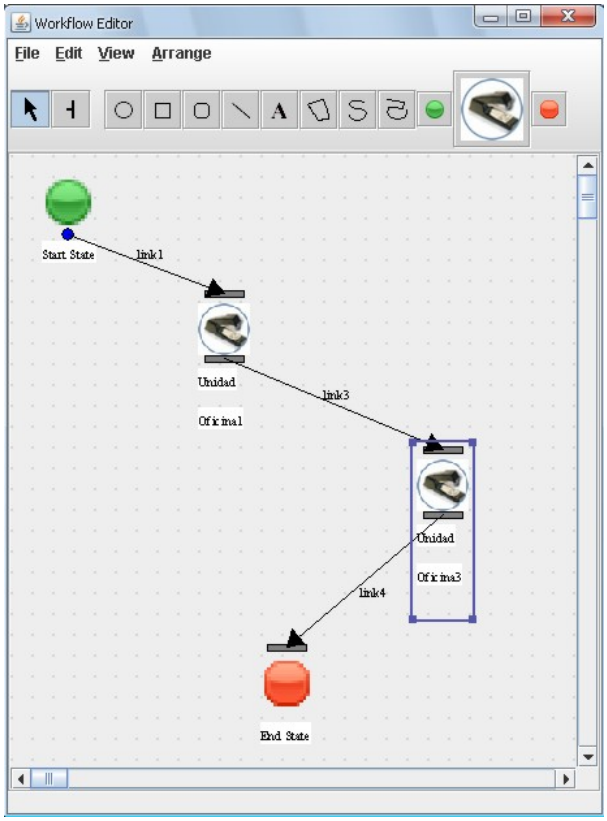
6- Vistas del editor MxGraph

GEF (Graphical Edition Framework) [17]

Es una librería Java [19] para desarrollar aplicaciones con un editor gráfico. Está basado en nodos, puertos y arcos, con los que se construye un flujo. Para esto se debe usar extensiones de las clases bases (NetNode, NetPort, NetArc).

Basándose en editores por defecto, se crea una aplicación (puede ser un applet) donde se le agregan botones, imágenes y formularios, para introducir la lógica necesaria para la integración de ésta herramienta con la aplicación. Para probar esta solución, se crearon tres nodos: el Nodo Inicial, el Nodo Unidad y el Nodo Final. En el Nodo Unidad se le agrega la lógica que se necesite para la solución del sistema, por ejemplo el nombre de la unidad y su ubicación.

En la figura 7 se muestra cómo quedaría un flujo usando este editor.



7- Definición de un workflow gráficamente en GEF

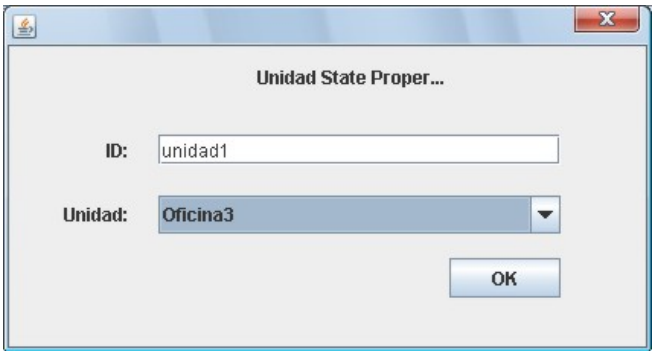
En cada nodo se le puede agregar datos que sean relevantes para la solución buscada. Para la demostración se le agregó una lista de Oficinas. En la figura 8 se muestran las propiedades del nodo Unidad.

Luego de generar el flujo con el editor, se agregaría una acción que a partir del dibujo genere una estructura que cumpla con el estándar XPDL, que definirá el flujo para el sistema.

A la definición se le pueden agregar nodos, por ejemplo para las decisiones. Otra extensión posible es la de agregar distintas propiedades a las aristas.

Esta solución tiene varias contras. Esta librería no tiene una muy buena documentación, y no es muy fácil de entender, ya que contiene una gran

cantidad de clases. Además no se encontraron muchos ejemplos de su utilización.



8- Vista del cuadro de dialogo de propiedades de un nodo en GEF

A continuación se evalúan las ventajas y desventajas generales de la opción de editor integrado antes descripta.

Ventajas:

- Es de código abierto.
- Se encuentra integrado con el sistema. Esto permite una comunicación más fluida y permite utilizar el flujo del negocio.
- Mayor usabilidad para el usuario, validación al momento del diseño.

Desventajas:

- Gran parte de la solución debe ser implementada, su evolución y testing estarán limitados a la del resto del sistema y especialmente a la dedicación sobre este punto.
- Mayor esfuerzo de implementación y cumplimiento de un estándar.
- Solución más limitada a nivel de workflow definibles, se ofrecerán al usuario solo acciones válidas.

COMPARACIÓN DE LAS ALTERNATIVAS

En esta sección se comparan las alternativas antes mencionadas priorizando a las distintas ventajas y desventajas, teniendo en cuenta las siguientes cualidades del software:

- *Integración*

  La integración de una herramienta externa con el sistema tiene una alta complejidad adicional. Y la misma está limitada en la cantidad de lógica del negocio que se le puede incorporar para el diseño del workflow. Por ejemplo, no se podría limitar la interacción entre las unidades. Por lo que, la validación del diagrama se realiza luego de finalizada la definición de la ruta.

Este punto puede ser crítico a la hora de la aceptación de los usuarios, aunque estos serán usuarios específicos para realizar esta funcionalidad dentro de la organización y tendrán una capacitación especial para el uso de la herramienta.

Estos problemas no existirían en el caso de que la edición de estos diagramas sea ofrecida por el mismo sistema. Ya que la integración es parte de la implementación y la obtención de información del negocio no es una restricción. La validación se podría realizar al momento de ir definiendo la ruta.

- *Usabilidad*

  Las herramientas externas ofrecen muchas más funcionalidades de las necesarias para la definición de las rutas de unidades, las que podrían confundir al usuario. En esta herramienta externa el usuario debe salir de la aplicación para la generación del diseño, saliendo así del contexto.

  La validación luego de finalizado el diagrama, puede provocar frustraciones en el usuario y un gran retardo en la generación de las rutas de unidades.

  En cuanto a una solución ofrecida por el mismo sistema, depende de la implementación que se realice, pero como ventaja frente a la anterior, podría no ser necesario acceder a una aplicación externa y sería más específica, por lo que no tendría limitaciones y no ofrecería funcionalidades innecesarias para el usuario.

- *Extensibilidad*

  En cuanto a la utilización de un editor externo, la extensibilidad de la aplicación se puede ver acotada si en determinado momento se quiere agregar alguna funcionalidad que ese editor no ofrece o que no se puede integrar, en cambio, estos editores generalmente ofrecen todo lo que un workflow necesita.

  La extensibilidad del editor implementado dentro de la aplicación va a depender en parte de la extensibilidad del sistema completo.

- *Esfuerzo de implementación y mantenimiento*

  En el uso de una herramienta externa no habría esfuerzo de implementación ni de mantenimiento, el esfuerzo estaría en la integración entre esta herramienta y el sistema.

  Los esfuerzos de implementación y mantenimiento utilizando un editor integrado son mucho mayores a los esfuerzos que implica la integración con una herramienta externa. A este se le agrega el esfuerzo de análisis, diseño y testing.

- *Evolución*

  Una aplicación externa evoluciona por sí misma y el sistema tendría que adaptarse a estos cambios, si se entiende que es necesario o si se modifican funcionalidades que lo afecten directamente.

  La evolución de un sistema integrado depende del diseño y de la implementación antes realizada.

3.3.3. Ejecución de Workflow – Motores

Para la ejecución de los workflows se utilizan los denominados motores de workflow. Un motor de workflow resuelve la comunicación entre los procesos diseñados y la ejecución de los mismos. En este caso, la comunicación entre el diagrama de rutas y su ejecución o entre los estados de las formas documentales y su ejecución, es el software que provee el entorno de ejecución para las instancias de los procesos de workflow.

Como en la parte de la definición de rutas, se tienen dos alternativas para este punto. Se puede utilizar un motor externo o que la aplicación provea uno. Además estos motores pueden utilizar un estándar para la definición de los workflows o no.

Como se explicó al inicio de este punto, existen varios requerimientos de workflows. Los requerimientos de cambio de estado de los documentos y los de manejo de éstos dentro de una unidad son más estables y no requieren ser definidos por los usuarios. En cambio para la definición de las rutas que van a realizar las distintas formas documentales son definibles por los usuarios.

Para la ejecución del workflow de rutas, se puede utilizar un motor externo o no. Además como se explicó en la sección anterior, la comunicación entre las cajas negras (Definición y Ejecución) es por medio de una estructura. Ésta estructura puede ser un estándar de definición de procesos (XPDL, BPEL).

Los workflows de cambios de estado y manejo de documentos son básicamente un diagrama estable de estados y no son definibles por los usuarios.

A continuación se analizan las distintas posibilidades de motores que se pueden utilizar.

MOTORES EXTERNOS QUE UTILIZAN UN ESTÁNDAR

En esta sección se describen algunos motores existentes para la ejecución de workflows que se basan en estándares como XPDL o BPEL.

Enhydra Shark [15]

Es un motor de workflow que ejecuta sobre procesos basados en definición XPDL. Posee una librería que puede ser embebida en WEB / Swing / Aplicación de Consola, o puede ser desplegada como un web service, servicio EJB, servicio RMI, etc.

Ofrece soporte automático, manual o mixto de procesos workflow.

Se puede realizar un cambio fácil de la base de datos utilizada para la persistencia a través de DODS (Data Object Design Studio). Incluye secuencias de comandos para crear las tablas de DB2 [20], HSQL [21], MySQL [22], Oracle [23], PostgreSQL [39].

Soporta Tool Agent, concepto definido por WfMC [25]. Tool Agent proporciona un mecanismo general para la invocación de la aplicación independientemente de las facilidades de manejo de cualquier sistema de gestión de workflow.

Puede customizar tanto clases Java como variables.

Definición de Proceso:
- Para cada definición XPDL hay una fábrica que crea instancias de procesos.
- Cada instancia de proceso tiene un identificador único.

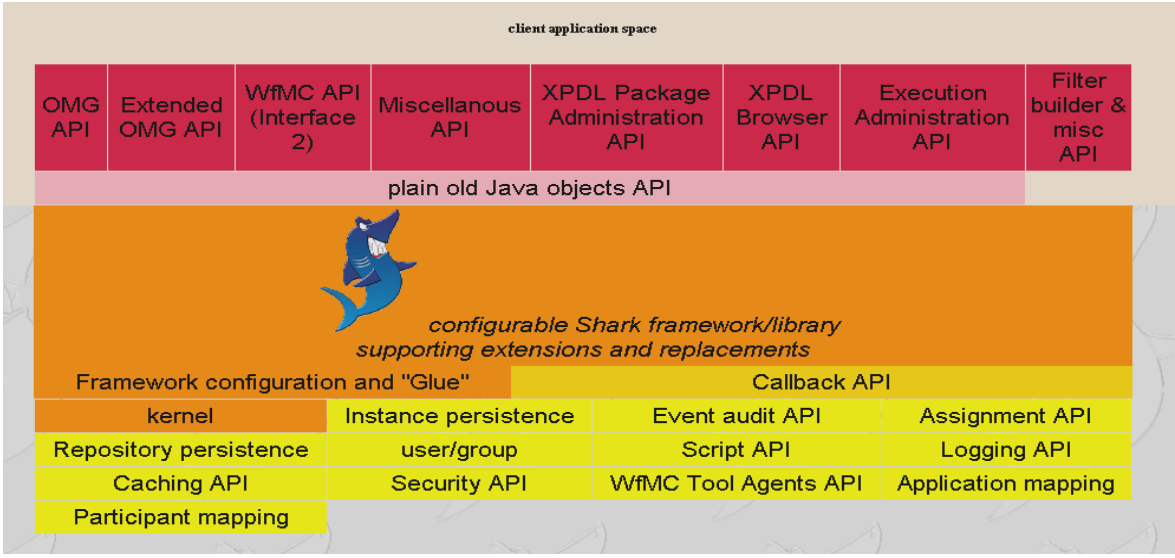
Definición de actividad:
- Creada durante el proceso de actuación y basada en el flujo y reglas definidas en el XPDL.
- Cada instancia de actividad tiene un identificador único.

Definición de Variables:
- Creadas durante el proceso de actuación y basada en la definición XPDL.
- El mismo Id que en el XPDL.
- Asignada al proceso o actividad.
- El valor va cambiando en tiempo de ejecución a través de la API o durante la ejecución del tool agent.

El motor ejecuta procesos basados en la definición de XPDL.

Basándose en los datos el motor determina cual es la próxima actividad que se llevará a cabo.

La arquitectura está dividida en tres componentes principales de la API: API cliente shark, shark kernel, y shark plugins API. En la figura 9 se muestra esta arquitectura.



9- Arquitectura de la API de Enhydra Shark

Intalio [26]

Intalio|BPMS es una herramienta open-source, que ejecuta el estándar BPEL. Está integrada por tres componentes, el Intalio|Designer el cual por medio de un entorno Eclipse [33] permite generar el diseño de un modelo BPMN y convertirlo en un proceso ejecutable, un segundo componente llamado Intalio|Server el cual es el encargado de la ejecución del workflow del mismo (es nativo BPEL 2.0, un servidor basado en J2EE [27]) y como tercer componente está Intalio|Workflow que es un workflow basado en BPEL4People [34] y es compatible con cualquier portal JSR 168 [35]. Intalio|Workflow soporta la ejecución de actividades BPEL4People a través de AJAX, la interfaz de usuario es generada automáticamente con XForms [36].

MOTORES EXTERNOS QUE NO UTILIZAN UN ESTÁNDAR

En esta sección se describe uno de los motores de workflow más utilizado, pero que no se basa en estándares, sino en un lenguaje propio.

JBoss jBPM [37]

Es un sistema flexible y extensible de administración de flujo de trabajo. JBoss jBPM cuenta con un lenguaje de proceso intuitivo para expresar gráficamente procesos de negocio en términos de tareas, estados de espera para comunicación asíncrona, temporizadores, acciones automatizadas, etc.

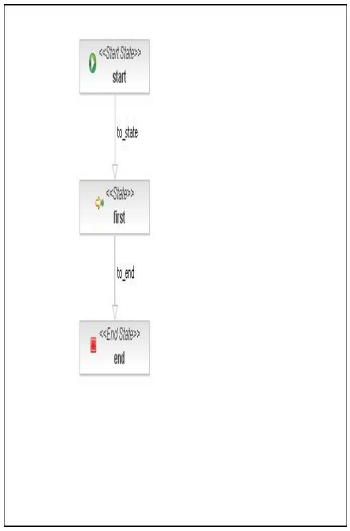
JBoss jBPM se puede configurar con cualquier base de datos y se puede implementar en cualquier servidor de aplicación.

El flujo de trabajo central y la funcionalidad BPM tienen un simple formato de biblioteca java. Esta biblioteca incluye un servicio para almacenar, actualizar y recuperar información de proceso de la base de datos jBPM.

JBoss jBPM también incluye una herramienta gráfica de diseño.

JBoss jBPM utiliza para la definición de proceso un esquema XML llamado JPDL.

A continuación se muestra un ejemplo de un diagrama de estados utilizando JBoss jBPM, en primer lugar gráficamente y luego el JPDL correspondiente.



10- Diagrama de estados utilizando JBoss jBPM

```
<process-definition xmlns="urn:jbpm.org:jpdl-3.2" name="simple">
   <start-state name="start">
      <transition name="to_state" to="first">
         <action name="action" class="com.sample.action.MessageActionHandler">
            <message>Going to the first state!</message>
         </action>
      </transition>
   </start-state>
   <state name="first">
            <transition name="to_end" to="end"></transition>
   </state>
   <end-state name="end"></end-state>
</process-definition>
```

Como con toda herramienta externa lo que puede llevar a problemas, es la integración a la aplicación.

Para el manejo de estados que se plantea para las formas documentales, la aplicación se comunicaría con el motor jBPM y le pediría la información de en qué estado se encuentra, y las posibles transiciones desde ese estado. Además se le podría pasar algún tipo de información para que quede en el histórico del workflow.

Ventajas:

- Se puede ejecutar un proceso en forma muy simple (mediante una API java).
- La conexión a la base de datos también es simple.

Desventajas:

- No maneja un estándar.

MOTOR INTERNO

La realización del motor en forma interna implica que sea desarrollado junto con el sistema. Lo ideal sería manejar un estándar en la definición del proceso y ejecutar este estándar en el motor de la aplicación. Esto permitiría algún día cambiar este motor por uno ya existente que se base en el estándar, como así también cambiar la herramienta que genera la definición del proceso en un estándar.

Esta solución requiere de mucho tiempo (de estudio y de implementación) para su realización y se podría decir que se estaría "reinventando la rueda", ya que existen varias herramientas que se encargan de la ejecución de workflows. Es más específica que las herramientas externas, ya que estaría dedicada íntegramente a la ejecución de este tipo de workflows. En cambio, las herramientas externas tienen el problema de la integración, sobre todo con soluciones específicas. Además, estas herramientas estarían abarcando mucho más de lo que se necesita en la solución.

COMPARACIÓN ENTRE LAS ALTERNATIVAS

El utilizar los motores ya existentes daría un gran poder para el manejo del histórico de los workflows y una gran eficacia, ya que están abocados a eso. Pero como se comentó anteriormente, la integración (comunicación) entre este motor y el sistema, puede llegar a ser muy complicado y se podría perder el gran potencial de este.

La realización del motor en forma interna, puede llegar a ser muy complicado y llevar mucho tiempo. Aunque se podría simplemente realizar una simplificación de un motor, que abarque los requerimientos que se precisan para esta solución (un diagrama de estados y una definición de rutas).

Otra opción sería hacer un híbrido. Manejar en el sistema el motor pero en el fondo utilizar un motor ya existente, que sirva para guardar el historial y para alguna otra cosa.

3.3.4. Conclusiones

Luego de todo este análisis y de conversaciones con el cliente, evaluando ventajas, desventajas y requerimientos, teniendo en cuenta que se quiere enfocar el proyecto como estudio de nuevas alternativas pero ofreciendo la posibilidad de cambiar algunas tecnologías por otras más adelante, es que se le da prioridad al uso de estándares.

Llegamos a una primera idea de usar un editor integrado con la aplicación que genere el estándar XPDL, lo cual hace a la edición de la ruta, independiente de la aplicación, pudiendo cambiar el componente elegido sin que esto afecte al resto del sistema. Esta decisión fue basada principalmente pensando en el usuario, por lo especificado en la comparación de las alternativas.

La opción elegida, para el alcance del proyecto, es el uso de la herramienta mxGraph [16]. A pesar de que no es un software libre, el cliente aceptó ésta opción con la condición del uso de ese estándar.

La elección de mxGraph [16] sobre GEF [17], que es la otra opción dentro de editores integrados, se da porque se ve una mayor facilidad para el manejo del código, que es abierto y en Java, JavaScript y XML, lenguajes bien conocidos. Mientras que GEF posee una API no muy fácil de entender y con poca documentación, como se explicó antes. Teniendo en cuenta el corto tiempo que se tiene para la implementación del prototipo, esta es la mejor opción para lograr la utilización de un editor gráfico que ayude a la creación de las rutas de unidades y del cambio de estado.

Para el caso del motor de workflow se decidió no utilizar un motor externo, sino realizar una implementación de las necesidades para la ejecución de las rutas de los documentos y del cambio de estado. Esta elección se realizó debido a que los motores estudiados ofrecen demasiadas funcionalidades y agregan complejidad a la solución sin aportar tantos beneficios que justifiquen la integración.

3.4. Persistencia – Bases de datos

En esta sección se estudian las diferentes formas en que se pueden persistir los documentos y la información que manejará el sistema.

Se tienen dos grandes áreas en cuanto a la persistencia de información para este proyecto, por un lado, las distintas formas documentales se manejarán en formato XML, este es un requerimiento no funcional, y por otro lado se debe persistir toda la información necesaria del sistema, como por ejemplo, la estructura organizacional. A su vez, las formas documentales pueden incluir datos adjuntos en otro formato (.doc, .pdf, .xls, .jpg, .avi, etc.), el cual también debe ser persistido y manejado por el sistema.

En los requerimientos planteados anteriormente se detectan tres alternativas de bases de datos a utilizar para la persistencia: bases de datos basadas en XML, bases de datos relacionales o un hibrido de ambas. Es importante analizar para cada alternativa las ventajas y restricciones que tienen para satisfacer los requerimientos de búsqueda full-text y almacenamiento de archivos adjuntos.

A continuación se describen las distintas alternativas planteadas y se comparan.

3.4.1. Bases de datos XML

En este punto se encuentran dos grandes ramas, las bases de datos nativas XML y las bases de datos que soportan XML (XML Enabled Databases).

Hay una diversidad de estrategias de almacenamiento XML, procesos de conversión y niveles de soporte para XML con los productos líderes de bases de datos.

Los documentos y los requerimientos de almacenamiento de datos XML tienen dos categorías generales:

- *Centrados en los datos (data-centric)*

  Usados para el transporte de datos. Son documentos XML con una estructura bien definida y contienen datos actualizables, pero de tamaño limitado y con reglas poco flexibles para campos opcionales y contenido. Para realizar una consulta se pueden tener datos estructurados que pueden extraerse del documento e indexarse con alguna base de datos convencional.

- *Centrados en el documento (document-centric)*

  Tienen una estructura irregular importante, tienden a ser más impredecibles en tamaño, con tipos de datos de tamaño limitado y reglas menos flexibles para campos opcionales y contenido. Realizar una consulta no es trivial, ya que se pretende hacer consultas no solo sobre el contenido, sino también sobre la estructura del documento.

De modo que una base de datos XML es aquella que define un modelo lógico de un documento de este formato, almacenando y recuperando documentos de acuerdo a ese modelo.

Luego de una introducción a las bases de datos XML, analizamos las dos grandes ramas, las bases de datos XML Nativas y las bases de datos con extensiones para soportar XML. Para más información sobre bases de datos específicas de cada rama ver los anexos B.3 y B.4 respectivamente.

BASES DE DATOS XML NATIVAS

XML Native Databases (Bases de datos nativas de XML): son aquellas que respetan la estructura del documento, se pueden hacer consultas sobre dicha estructura y es posible recuperar el documento tal como fue insertado originalmente. Son bases de datos centradas en documentos.

No existe una definición estándar de una base de datos nativa en XML pero la organización XML:DB Initiative [40] para bases de datos XML, describe una base de datos de este tipo como: "modelo lógico para documentos XML que almacena y recupera documentos de acuerdo a dicho modelo" [41].

Estas bases de datos surgen por la necesidad de gestión eficiente de grandes cantidades de documentos XML. Las empresas argumentan que: "los documentos XML no se pueden almacenar en sistemas de gestión de bases de datos convencionales por su naturaleza jerárquica y semi-estructurada".

Los esquemas XML son implementados en bases de datos XML nativas para registrar reglas de almacenamiento e indexación de datos y para proveer y obtener información de almacenamiento a los mecanismos de bases de datos XML nativas. Adicionalmente, todos los objetos en una base de datos XML nativa son típicamente accesibles directamente mediante un URL.

El trabajo con bases de datos XML nativas involucra dos pasos básicos:

- Describir los datos mediante Definiciones de Tipos de Datos (Document Type Definitions, DTD) o esquemas XML.
- Definir un nuevo esquema de base de datos XML nativa, XML o Mapa de Datos a usar para almacenamiento y obtención de datos.

Son bases de datos, y como tales soportan transacciones, acceso multi-usuario, seguridad, la API de programación, lenguajes de consulta, etc., diseñadas especialmente para almacenar documentos XML. La única diferencia con otras bases de datos es que su modelo interno se basa en XML y no otra cosa, como el modelo relacional. El aspecto principal de los productos es el almacenamiento de los documentos XML.

Alguna de sus características incluyen la validación de los documentos y soporte para varios lenguajes de consultas (DOM (Document Object Model) [44], XPath [42], XQuery [46]). Muchas bases de datos permiten realizar búsquedas utilizando full-text, también soportan el XUpdate para actualizaciones y borrados.

En las bases de datos XML nativas se manejan los cambios más fácilmente que en las bases de datos relacionales. Ofrecen mayor flexibilidad.

Dependiendo de la forma en que almacena los datos físicamente, podría ser capaz de recuperar datos más rápido que una base de datos relacional. La razón es que físicamente la información está toda junta y es más fácil recuperarla que buscando las relaciones lógicas de una tabla relacional. La

desventaja es que el aumento de velocidad solo se aplica cuando la recuperación de datos es en el orden en que se almacenaron en disco.

Otros usos de estas bases de datos incluyen el suministro de datos y metadatos para las caches de largas transacciones, el manejo de grandes documentos, la manipulación de datos jerárquicos.

El tipo de las bases de datos XML nativas se define en función de las diferentes estrategias de almacenamiento para los documentos. En este sentido se pueden destacar:

- *Almacenamiento basado en texto*

  Almacena el XML entero en forma de texto, y proporciona alguna funcionalidad de bases de datos para acceder hacia él.

  Se pueden aplicar técnicas de compresión para reducir el espacio de almacenamiento, se mantienen índices adicionales para aumentar la eficiencia en el acceso a la información. Pueden definirse sobre bases de datos o sobre sistemas de archivos.

- *Almacenamiento basado en modelo*

  Definen un modelo de datos lógico (DOM) para la estructura jerárquica de los documentos XML. Almacenan los documentos de acuerdo con este modelo, usando el modelo físico que se desee. Así resulta muy sencillo identificar y extraer información basándose en la estructura del documento.

- *Soluciones desarrolladas específicamente para la gestión de documentos XML.*

EXTENSIONES DE BASES DE DATOS PARA XML

XML Enabled Databases (Bases de datos habilitadas para XML): son aquellas que desglosan la información de un documento XML en su correspondiente esquema relacional o de objetos. Contienen extensiones para transferir datos entre documentos XML y sus propias estructuras de datos. Son usadas para aplicaciones centradas en los datos (data-centric), excepto cuando la base de datos también soporta almacenamiento XML nativo.

Son bases de datos relacionales que siguen almacenando toda la información de manera relacional, es decir en forma tabular (tablas, registros y columnas) o en caso contrario almacenan todo el documento en formato Binary Large Object (BLOB), pero la principal característica que brindan estas bases de datos es la capacidad de obtener los resultados de las consultas en formato XML; es por ello que dichas bases de datos pertenecen a la categoría de "XML-enabled database".

La diferencia entre bases de datos XML-enabled y nativas XML es que la primera utiliza el esquema de estructuras específicas (schema-specific structures) que deben ser mapeadas con el documento XML en tiempo de diseño.

Algunos productos de middleware y bases de datos XML-enabled soportan lenguajes de consulta XML, aunque generalmente sobre datos guardados en lugar de documentos XML.

Se pueden numerar diferentes formas de almacenamiento de los datos XML en este tipo de bases de datos:

- *Almacenamiento no estructurado*

  Almacenamiento del documento XML directamente en formato de texto en un CLOB (Character Large OBject). Lo soportan la mayoría de los sistemas de gestión de bases de datos relacionales. Incluyen además funciones para acceder el contenido de los documentos del SQL. Algunos ejemplos son: Oracle XML DB [23], IBM DB2 XML Extender [20], Microsoft SQLXML [24].

- *Almacenamiento estructurado*

  Un metamodelo detallado de documentos XML capaz de representar árboles de nodos de documentos XML arbitrarios, se construye utilizando primitivas de modelado del sistema de gestión de bases de datos convencional que hay por debajo. Los contenidos de los documentos XML se pueden consultar utilizando las facilidades proporcionadas por el sistema de gestión de bases de datos.

  Algunos ejemplos de productos son: XML Cartridge de Oracle [23], ozone/XML [47] (es una librería para clases persistentes OO ozone que implementa el estándar DOM para la representación de documentos XML en una base de datos).

- *Mapeo*

  El contenido de documentos XML se mapea en esquemas de bases de datos específicamente diseñado para este contenido. Permite utilizar las capacidades de modelado de los sistemas de gestión de bases de datos convencionales para la representación eficiente y adecuada del contenido de los documentos. Existen gran cantidad de herramientas y formalismos para la especificación del mapeo entre un formato XML y un esquema de base de datos. Mucha investigación respecto a la generación automática de esquemas de bases de datos relacionales a partir de documentos XML y el mapeo automático entre los mismos. Ejemplo: Oracle XML DB [23].

3.4.2. Bases de datos relacionales

Las bases de datos relacionales es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Este tipo de bases de datos son centrados en los datos, la información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información. El lenguaje más habitual para construir las consultas a bases de datos relacionales es SQL (Structured Query Language), un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

Como se dijo antes, los documentos y los requerimientos de almacenamiento de datos XML se pueden almacenar centrándose en los documentos y no en los datos. Esto es posible en una base de datos relacional u orientada a objetos, debiendo realizar un traspaso del XML a tablas u objetos para el almacenamiento y el trabajo en viceversa para recuperar los datos en el formato XML.

Otros puntos importantes a analizar de una base de datos relacional, son la búsqueda full-text y el manejo de archivos adjuntos. Esto se analiza en los siguientes puntos.

FULL- TEXT

Las bases de datos relacionales poseen distintas formas de realizar la búsqueda full-text, a continuación se ejemplifican algunas.

OpenFTS (open-source full-text Search)

Es un motor de búsqueda basado en PostgreSQL [39] que proporciona la indexación de datos y el posicionamiento de relevancia para la búsqueda en la base de datos.

SQL Server Search

El Servicio Microsoft Search [48] es un motor de indexación y de búsqueda de texto que permite al SQL Server [38] realizar consultas de texto eficaces y sofisticadas sobre columnas que almacenan datos basados en caracteres.

Entre las diferencias existentes entre este servicio y las búsquedas que utilizan el operador LIKE de SQL [49], se pueden citar las siguientes:

- Se almacenan en el sistema de archivos y no en la base de datos, aunque es la base de datos quien los administra.
- Sólo se permite un índice de texto por cada tabla.
- Si se desea agregar datos a los índices de texto hay que realizar un llenado manual o programado (también se pueden llenar automáticamente en la inserción de datos).
- Son más rápidos y flexibles.

El índice que se crea sobre una columna de texto almacenará información sobre cada palabra que contiene la columna y su ubicación dentro de la tabla. Se pueden actualizar (como los índices normales del SQL Server) al modificar los datos de la tabla, o se pueden llenar a intervalos regulares. Estos procesos de llenado se suelen realizar de forma asíncrona y en segundo plano porque consumen tiempo y recursos.

PERSISTENCIA DE ADJUNTOS

Entre los principales requerimientos planteados se encuentra la necesidad de almacenar archivos adjuntos con la misma seguridad que posee el documento (ej. expediente, formulario, etc.) que lo contiene. Por lo tanto la mejor alternativa para asegurar el nivel de seguridad que se requiere es almacenar el archivo en la misma base de datos.

La forma de almacenar archivos adjuntos (como por ejemplo, pdf, doc, txt, jpg, etc.) en las bases de datos relacionales es transformando el archivo en binario y almacenarlo usando SQL [49]. Para conocer opiniones sobre esta forma de almacenamiento se recurrió a los foros, donde se menciona que puede generar una pérdida de performance en la base de datos y problemas de crecimiento de la misma, dependiendo del tamaño de los adjuntos. Además se tiene otro requerimiento a satisfacer, que es la búsqueda full-text dentro de estos archivos, lo que también hace perder performance almacenando el documento como binario. Como alternativa para estos problemas existen repositorios que permiten almacenar en bases de datos y ya ofrecen las funcionalidades de búsqueda full-text en los archivos, estos repositorios son estudiados en la sección 3.5.

3.4.3. Comparación BBDD XML con relacionales

A continuación se muestra una tabla donde se comparan las ventajas y desventajas de la utilización de cada una de las bases de datos.

	XML	**Relacionales**
Manejo de formas documentales XML – Metadatos	Almacenan el XML y devuelven el XML, no requieren de una transformación a XML para luego ser utilizado en la aplicación. Al manejar directamente el XML permite la flexibilidad necesaria para agregar y quitar nuevos metadatos a las diferentes formas documentales.	Se requiere de un procesamiento de los datos retornados para obtener el XML que representa. Las formas documentales no tienen una estructura definida y tienen requerimientos de incorporación de nuevos datos según las necesidades, para esto se debe realizar un diseño especial de la base de datos.

Archivos Adjuntos	Algunas bases de datos XML nativas permiten guardar archivos adjuntos en otros formatos y ofrecen búsqueda full-text en esos archivos.	Los archivos se guardan en formato binario, esto baja la performance y aumenta el tamaño de la base de datos porque no usa ningún sistema de compresión.
Full-text	Maneja indexación de XML y en las bases de datos donde se guardan documentos, también, lo que permite la búsqueda full-text.	Esta funcionalidad depende del motor que se utilice. Los motores más usados brindan la capacidad de indexación.
Performance	La búsqueda de información que se encuentra sobre el final de un archivo de gran tamaño baja la performance, pues tiene que recorrer todo el documento para encontrarla.	Para poder manejar metadatos es necesario tener varias columnas con valores nulos, o muchas tablas, lo que hace que baje la performance.
Persistencia de Objetos	Para los módulos donde la aplicación no maneja XML, se debe realizar la transformación de la información a XML y para obtener los datos, la transformación inversa.	Existen varios frameworks que se encargan de la persistencia de los objetos en la base (Hibernate, JPA, etc.).
Ventajas comerciales y de venta	Los usuarios sienten que las bases de datos XML nativas, no son bien conocidas o bien soportadas como ellos lo preferirían.	Los clientes principalmente tienen bases relacionales ya instaladas, es una estrategia comercial que el sistema pueda ser instalado utilizando las bases de datos que ya dispone el cliente. Esto es una gran ventaja para las organizaciones que ya tienen una administración establecida de las mismas.
Historia y futuro	Son relativamente modernas pero existe una gran tendencia en la utilización de XML en las aplicaciones.	Estas bases de datos se vienen utilizando desde hace mucho tiempo y se encuentran estables.

3.4.4. Conclusiones

En base a las comparaciones realizadas anteriormente entre los diferentes tipos de bases de datos, se decidió junto con el cliente tener dos alternativas y realizar un diseño que permita el cambio de una por la otra.

La primer alternativa es el uso de una base de datos relacional con extensión para XML, de esta forma se pueden manejar tanto los módulos relacionales como las formas documentales en una misma base de datos, unificando la seguridad de acceso a datos. Esta alternativa permite mezclar el mundo relacional y el XML, usando la ventaja de cada uno para satisfacer los requerimientos. El principal problema del uso de base de datos relacionales es la transformación de los documentos XML y la flexibilidad al agregar nuevos metadatos, esto se soluciona con las bases de datos relacionales extendidas a XML, porque permite almacenar los XML y consultarlos a través de XPath y/o XQuery.

De las opciones posibles para este tipo de base de datos se eligió PostgreSQL con extensión para XML [39], porque se encuentra dentro de las bases de datos open-source mas conocidas en el entorno.

Las bases de datos nativas XML fueron descartadas, porque para manejar los datos de los módulos relacionales es necesaria la transformación a XML y además se perderían funcionalidades necesarias, que sí incluyen las bases de datos relacionales. La mejor solución sería incluir dos bases de datos, una relacional y la otra XML nativa. Sin embargo, esto traería problemas de relacionamiento entre los datos y además incrementaría los gastos de administración.

Como segunda alternativa se planteó la utilización de una base de datos relacional para todos los módulos, teniendo que realizar la conversión del XML a la estructura de la base de datos.

Independientemente de la base de datos que se utilice es necesario utilizar un repositorio de archivos para los documentos adjuntos, ya que ninguna de las opciones ofrece un buen manejo de este tipo de documentos. Excepto una de las bases de datos XML nativa, que se muestra en el anexo B.3 (TEXTML [50]), que incluye un repositorio, pero se estaría atando la solución a un proveedor.

3.5. Repositorios de archivos

Para el almacenamiento de los archivos adjuntos se analiza la utilización de repositorios de archivos.

Java Content Repository API [51] (JCR) es una especificación para una plataforma Java API, para acceder a repositorios de contenido de manera uniforme. El contenido de los repositorios se utiliza en sistemas de gestión

de contenidos (CMS en inglés) para mantener los datos de contenido y también los metadatos utilizados en la CMS. La especificación fue desarrollada en el marco del Java Community Process como JSR-170 (versión 1) y como JSR-283 (versión 2).

A continuación se describe uno de los repositorios estudiados.

Apache Jackrabbit [30]

Es un repositorio de archivos open-source para la plataforma Java que implementa el estándar JCR. El proyecto se graduó en la Incubadora Apache el 15 de marzo de 2006, y ahora es un proyecto de nivel superior de la Apache Software Foundation [52].

Características:

- Es completo y plenamente compatible con la aplicación de la API Content Repository for Java Technology [51] (JCR) y, por lo tanto, su principal API se define por JCR. Para un desarrollador esto significa que la mayoría de las operaciones requeridas son definidas por el JCR API. Las clases e interfaces en Apache Jackrabbit sólo se necesitan cuando se accede a la funcionalidad que no se especifica en JCR.

- Implementa XPath y opcionalmente la sintaxis de consulta SQL. Su diseño sigue el objetivo de la JSR-170, especificación de que todos los elementos obligatorios de consulta se pueden expresar ya sea en XPath o en SQL [49]. De este modo, la actual implementación es independiente del motor de sintaxis de consulta utilizado, aunque las consultas internas de Jackrabbit están más cerca de XPath que de SQL, debido a la estructura jerárquica de un JCR [52].

- Utiliza Lucene [53] como el índice, para la búsqueda, subyacente de la aplicación y proporciona varias extensiones y personalizaciones que ayudan a mejorar el rendimiento en un entorno donde los cambios en el índice son frecuentes. Las extensiones también cubren las características que no son compatibles con Lucene, al igual que las consultas jerárquicas. El índice soporta varias extensiones de archivos (pdf, doc, xls, ppt, xml, html).

3.5.1. Conclusiones

Dentro de los repositorios estudiados el Apache Jackrabbit ofrece seguridad sobre los documentos y es eficiente para las búsquedas gracias al uso de Lucene, es por eso que es el elegido para utilizar en la implementación del prototipo.

3.6. Manejo de XML

La aplicación requiere de un fuerte manejo de XML, por lo que se considera necesaria la utilización de bibliotecas que faciliten la manipulación de datos XML.

Hoy en día existen muchas herramientas para la manipulación de XML en diversos lenguajes, la diferencia entre ellas es la metodología que utilizan para interactuar con los elementos XML. A continuación se detallan algunas.

DOM (Document Object Model) [44] y SAX (Simple API for XML) [45]

Mientras DOM trabaja con árboles de información, SAX lo hace a través de eventos.

El DOM, es un modelo computacional a través del cual los programadores pueden acceder y manipular dinámicamente el contenido, estructura y estilo de los documentos HTML y XML. DOM es del consorcio W3C (World Wide Consortium) [54].

SAX proporciona un mecanismo para la lectura de los datos de un documento XML. Se trata de una alternativa popular al DOM.

Un analizador que implementa SAX funciona como un analizador de flujo, con un evento impulsado por la API. El usuario define una serie de métodos que serán llamados cuando los acontecimientos se producen durante el parsing. Los eventos incluyen, texto nodos, elementos nodos, instrucciones de procedimientos de XML y comentarios. Los mismos son disparados cuando comienza cada una de estas características XML y de nuevo cuando finalizan.

Los analizadores SAX tienen ciertas ventajas sobre DOM, por ejemplo la cantidad de memoria que utiliza el analizador, porque los analizadores DOM deben tener todo el árbol en memoria para comenzar, entonces la cantidad de memoria que consume es mucho mayor que SAX, además de que la asignación de memoria lleva su tiempo.

JDOM [55]

Se encuentra entre las bibliotecas open-source. A pesar de su similitud con DOM, es una buena alternativa, su principal diferencia es que mientras DOM fue creado para ser un lenguaje neutral e inicialmente usado para manipulación de paginas HTML con JavaScript [18], JDOM se creó específicamente para usarse con Java y por lo tanto beneficiarse de sus características. JDOM proporciona una manera de representar a esos documentos para su fácil y eficiente lectura, manipulación y escritura. Tiene una API liviana y rápida.

3.6.1. Conclusiones

Uno de los requerimientos no funcionales es el uso del lenguaje Java para la implementación, es por eso que se decidió manejar los XML mediante la librería JDOM.

3.7. Presentación web - Prototipo

La presentación no es un punto fuerte en este proyecto de grado. Lo que se quiere es realizar una aplicación que permita probar el funcionamiento del sistema y la integración de los distintos elementos elegidos para utilizar en este sistema.

Para la realización del prototipo se necesita una herramienta web. En esta sección se analizan los distintos requisitos que debe contemplar la herramienta a utilizar para poder ser tenida en cuenta y se comenta la herramienta elegida.

Requisitos:

- Funcionalidad en los navegadores más conocidos.
- Manejo de JavaScript (para poder utilizar el editor integrado de rutas de workflow, MxGraph).
- Lenguaje Java.
- Ajax, para hacer una rica interfaz.

Cómo la aplicación no va a tener un gran tamaño, se optó por utilizar una herramienta que permita la creación rápida y eficiente de una aplicación Web. La Herramienta estudiada se llama ZK.

ZK [31]

Es un framework de aplicaciones web que utiliza la tecnología Ajax, completamente basado en Java, de código abierto que permite una rica interfaz de usuario para aplicaciones web con poca programación.

ZK es completamente personalizable y ampliable con módulos. Con el uso de CSS [57], plantillas, y componentes, la apariencia y el comportamiento puede ser reemplazado sin modificar drásticamente la aplicación.

Ventajas:

- Permite la realización de una rica interfaz sin mucho conocimiento previo.
- No es necesario saber Ajax ni JavaScript.
- Modelo basado en componentes (se permite la creación de nuevos componentes mediante clases Java).
- Funciona en la gran mayoría de navegadores (IE, Firefox, Safari, Opera).

- Corre en cualquier servidor Web que soporta Servlet 2.3 o mayor y JVM 1.4 o mayor.
- Existen plugins para Eclipse [33] y NetBeans [56].
- Integración con otros frameworks (JSP, JSF, Spring, Struts, Hibernate, etc.).

Desventajas:

- No utiliza un estándar como JSP o JSF.
- No utiliza Model View Controller (MVC).

3.7.1. Conclusiones

Por la gran facilidad que ofrece para implementar aplicaciones web y lo sencillo que es se decidió utilizar el framework ZK para la implementación de la presentación.

Capítulo 4 Diseño y arquitectura

4.1. Introducción

La arquitectura del sistema a desarrollar se define mostrando las diferentes vistas según el modelo 4+1, en el cuál se muestra la vista de casos de uso, la vista del modelo de diseño, la vista del modelo de implementación y la vista del modelo de distribución.

La *vista de casos de uso* muestra la funcionalidad del sistema como es percibida desde el exterior, así como también describe un conjunto de escenarios y casos de uso que tienen una cobertura arquitectónicamente significativa o que ilustran un punto específico de la arquitectura. Estos son los casos de uso relevantes a la arquitectura.

La *vista del modelo de diseño* describe el diseño más importante de las clases, su organización en paquetes y subsistemas, y la organización de estos en capas. También contiene algunas realizaciones de casos de uso. Muestra como la funcionalidad es diseñada en el interior del sistema, en términos de la estructura estática y comportamiento dinámico del sistema.

La *vista del modelo de implementación* muestra la organización del código. Contiene una visión general del modelo de implementación y su organización en términos de módulos en paquetes y capas. También se describe la asignación de paquetes y clases del modelo de diseño a los paquetes y módulos de la vista de implementación. Esta vista es opcional, ya que sólo se realiza en los casos donde la implementación no se conduce estrictamente por el diseño. En este caso el empaquetado de los modelos de diseño y de implementación son idénticos, es por esto que esta vista es omitida.

La vista *del modelo de distribución* describe varios nodos físicos para las configuraciones más típicas de las plataformas y la asignación de las tareas a los nodos físicos.

4.2. Alcance

Se definió el alcance para el diseño junto con el cliente, teniendo en cuenta los casos de usos más importantes y relevantes para definir la arquitectura.

Considerando que las distintas formas documentales son independientes y que la arquitectura debe permitir ingresar nuevas formas documentales, se decidió agregar las dos que aportan más a la arquitectura: Expediente y Formulario. Es imprescindible implementar del módulo gestión de documentos, pues es utilizado por todas las formas documentales, como también el módulo que se encargará de la ejecución del Workflow.

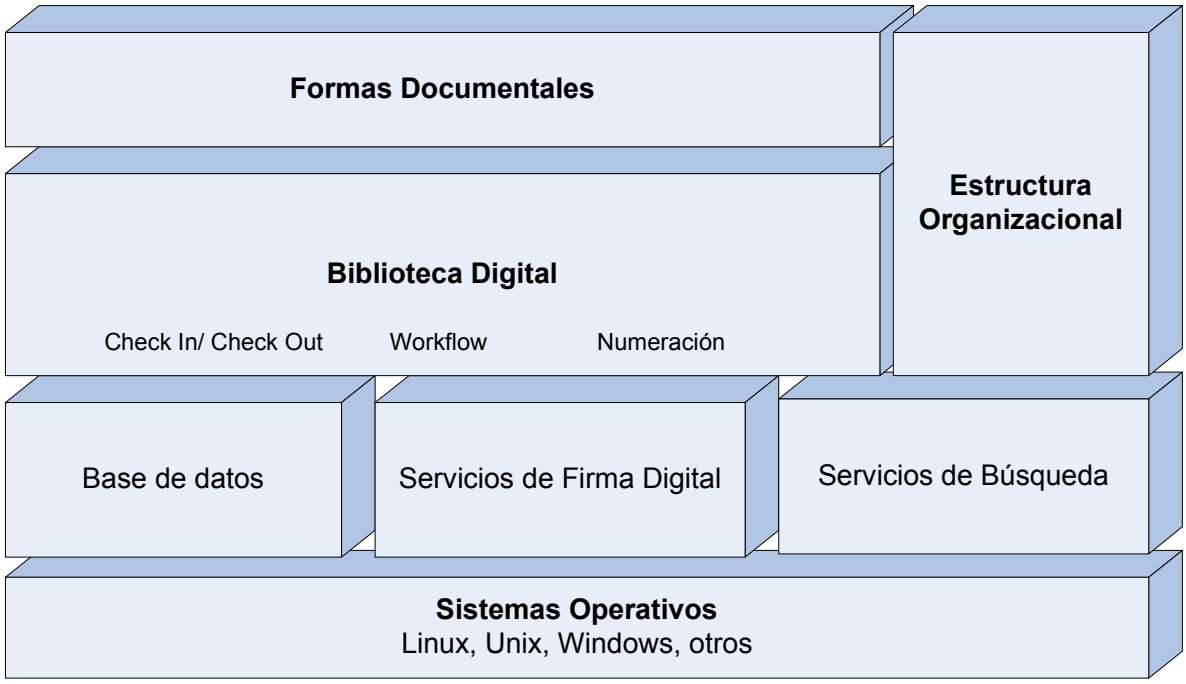
Por otro lado, casos de usos relevantes a la arquitectura se encuentran en la administración de usuarios, en las bandejas de documentos, en la estructura de la organización y en los requerimientos que competen a la seguridad del sistema.

Quedan por fuera del alcance entonces, los requerimientos de las formas documentales Comunicaciones y Resoluciones, las preferencias de los usuarios, la definición del calendario, estadísticas, reportes y auditoría.

4.3. Vista inicial de la arquitectura

4.3.1. Primera aproximación de la arquitectura

En esta primera vista de la arquitectura se tiene una arquitectura separada en capas. En la capa de más abajo se encuentran los sistemas operativos sobre los que tiene que funcionar el sistema. Luego, dentro del sistema, en la capa inferior se encuentra la base de datos, los servicios de la firma digital y los servicios de búsqueda. En la capa del medio se tiene la lógica del manejo de documentos que utiliza, entre otros, un motor de workflow, un módulo que se encarga de la numeración de las distintas formas documentales y un módulo que se encarga de la reserva de los documentos a través de check in / check out. En la capa superior se encuentran las distintas formas documentales (Expediente, Formulario,…). A su vez, a la altura de las dos capas de más arriba se encuentra la estructura de la organización.



11- Primer aproximación de la arquitectura

La figura 11 muestra una primera aproximación de la arquitectura, teniendo en cuenta a grandes rasgos los requerimientos funcionales y no funcionales.

4.3.2. Modelo de Dominio

En esta sección se presentan los principales conceptos del dominio del problema que se está modelando así como la relación que existe entre ellos. Se incluyen diagramas de modelo de dominio expresando gráficamente estos conceptos y relaciones. Estos diagramas se separaron por módulos, al igual que el documento de requerimientos. Se muestran los modelos más interesantes desde el punto de vista del diseño de la aplicación.

EXPEDIENTE

Expedientes es una de las formas documentales existentes.

Se realizó un modelo de dominio que muestra gráficamente los componentes de un expediente y sus asociaciones.

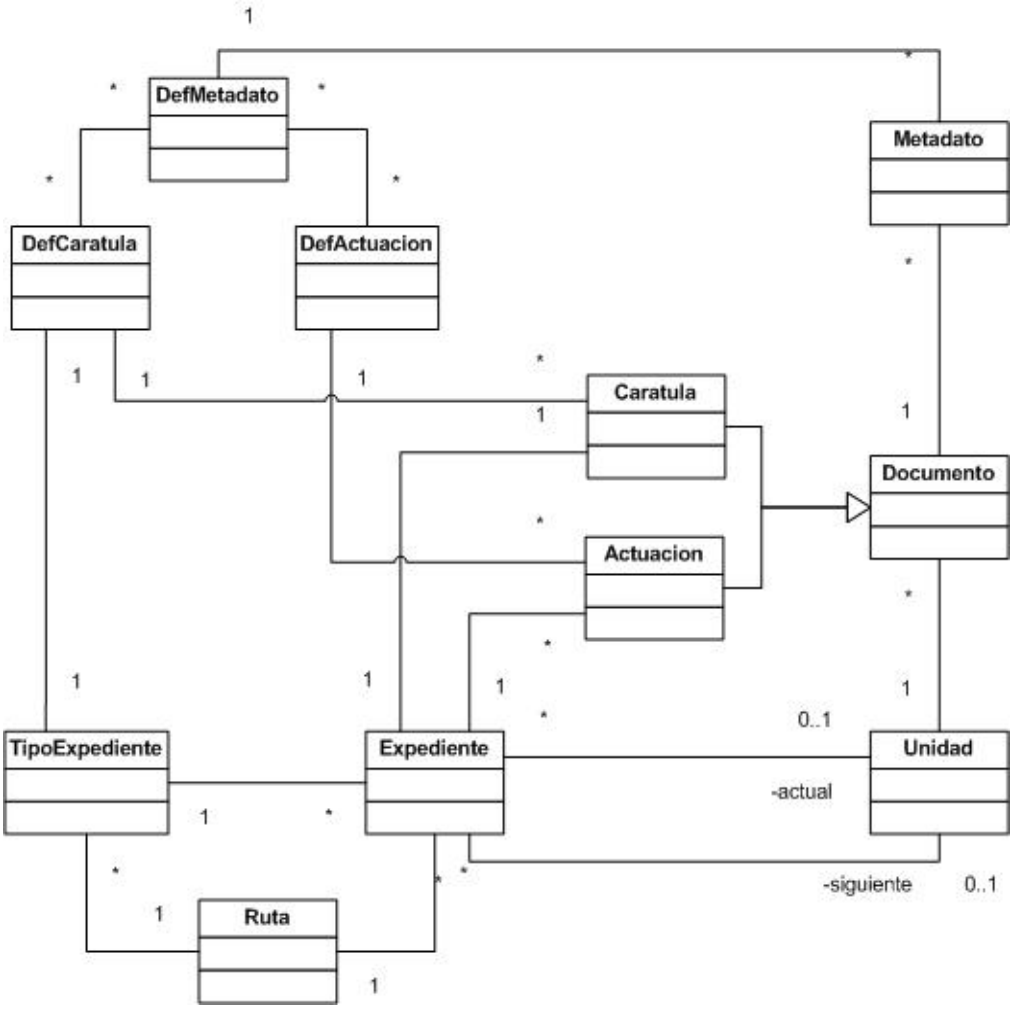
Un expediente está formado por un documento carátula y varios documentos actuaciones, estos últimos mantienen un orden.

Existen distintos tipos de expedientes que definen los datos de la carátula y la ruta de unidades por la cual pueden pasar los expedientes de ese tipo. Los documentos (actuación o carátula) se encuentran en una unidad y tienen varios metadatos.

Lista de entidades detectadas:

- Expediente: representa el trámite expediente.
- Documento: representa la entidad documento. Estos documentos van a ser manejados por la biblioteca documental (ver sección sobre el Gestor Documental).
    - Carátula: representa el documento carátula del expediente.
    - Actuación: representa el documento actuación del expediente.
- Unidad: representa una unidad u oficina de la organización.
- Tipo de Expediente: los tipos de expedientes existentes. Los tipos de expedientes tienen versiones, al crear un expediente de un tipo se le asigna la última versión de ese tipo.
- Ruta: workflow de unidades por la que pasa el expediente.
- Definición de la Carátula: metadatos que puede poseer el documento carátula.
- Definición de la Actuación: metadatos que puede poseer el documento actuación.
- Definición de Metadato: descripción del metadato para ser presentado al usuario.
- Metadato: nombre y valor de un dato de un documento.

En la figura 12 se muestra el modelo de dominio para Expedientes, mostrando gráficamente las diferentes entidades y sus asociaciones.



12- Modelo de dominio expediente

Este modelo tiene algunas restricciones de dominio entre las cuales se destacan las siguientes:

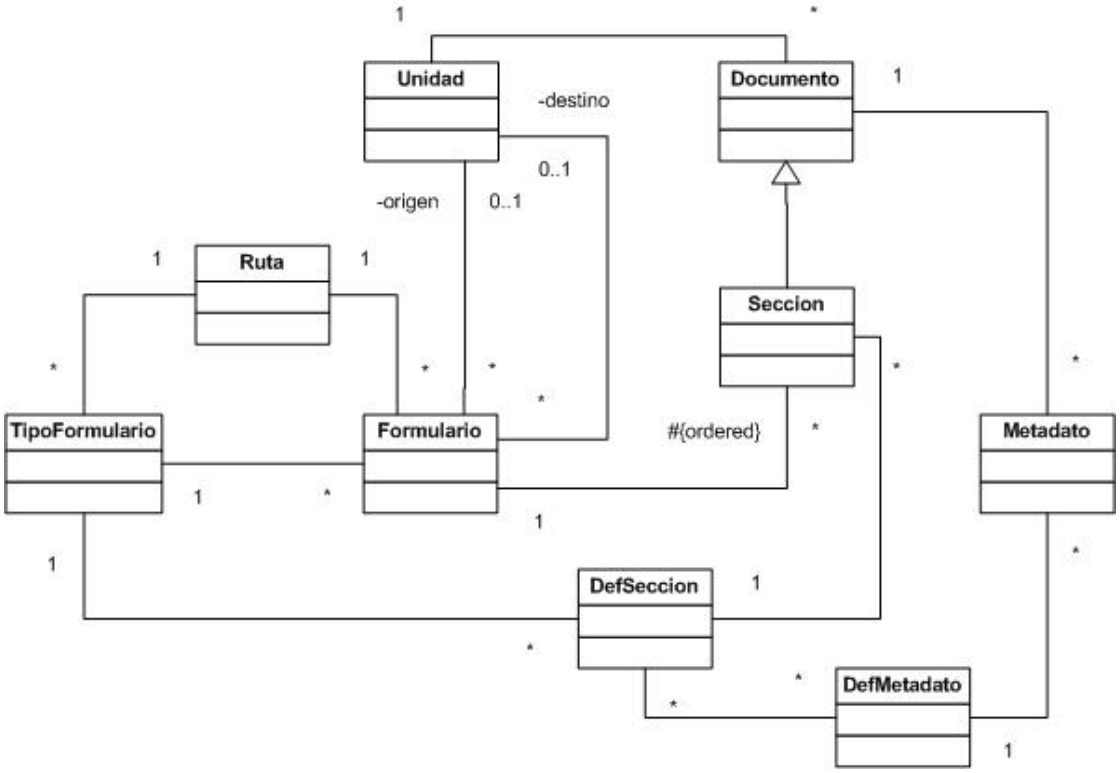
- Cuando se crea un nuevo expediente de un tipo, ese expediente tiene la carátula con la definición de la caratula asignada al tipo de expediente al momento de creación del mismo.
- La ruta del expediente, es la ruta que tiene definida el tipo de expediente al momento de la creación.
- La unidad de la caratula es una de las unidades creadoras del expediente y una de las unidades que pueden iniciar el workflow de la ruta.
- Si un documento tiene un metadato la definición de ese metadato pertenece a la definición del documento.

FORMULARIO

Formulario es otra de las formas documentales.

Un formulario, a diferencia de los expedientes está formado por una cantidad fija de documentos llamados sección.

Existen distintos tipos de formularios que definen los datos para cada una de sus secciones. Los documentos sección se encuentran en una unidad y tienen varios metadatos.



13- Modelo de dominio formulario

Lista de entidades detectadas:

- Formulario: representa el trámite formulario.
- Documento: representa la entidad documento. Estos documentos van a ser manejados por la biblioteca documental (ver módulo Gestor Documental).
    - Sección: representa el documento sección del formulario.
- Unidad: representa una unidad u oficina de la organización.
- Tipo de Formulario: los tipos de formularios existentes. Estos tipos tienen versiones, al crear un formulario de un tipo se le asigna la última versión de ese tipo.
- Ruta: workflow de unidades por la que pasa el formulario.
- Definición de la sección: metadatos que puede poseer el documento sección.
- Definición de metadato: descripción del metadato para ser presentado al usuario.
- Metadato: nombre y valor de los datos de un documento.

El la figura 13 se muestra el modelo de dominio para formularios.

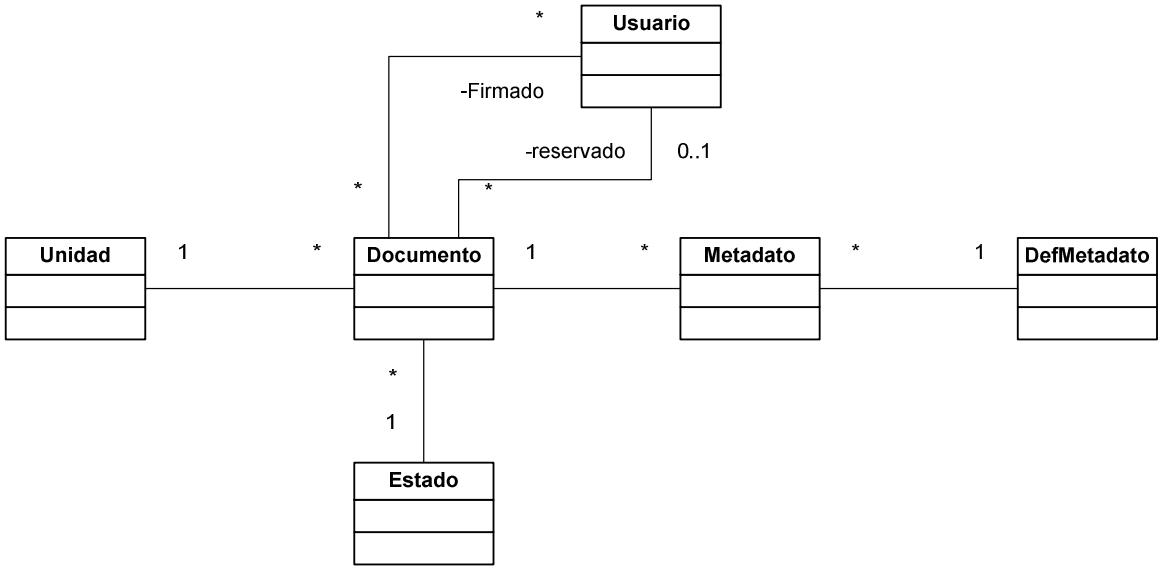
Sobre este modelo de dominio se realizan las siguientes restricciones:

- Los formularios de un tipo tienen menos o igual cantidad de documentos secciones como definiciones de secciones tiene el tipo.
- Las secciones de un formulario mantienen el mismo orden que el orden de las definiciones de secciones que tenía el tipo de formulario cuando se creó este formulario. La sección n de un formulario tiene como definición la definición de sección n que tenía el tipo de formulario cuando se creó el formulario.
- La ruta definida para el tipo de formulario tiene una cantidad igual o menor a la cantidad de secciones de ese tipo.

GESTOR DOCUMENTAL

Cuando se relevaron los requerimientos se vio la necesidad de tener una biblioteca documental que ofreciera los servicios básicos sobre los documentos.

En esta sección del documento se analizan los conceptos que debe manejar esta biblioteca y se realiza el diagrama de dominio para la misma.



14- Modelo de dominio de biblioteca documental

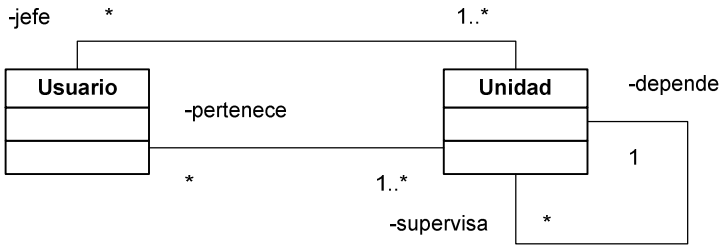
Lista de entidades detectadas:

- Documento: representa la entidad documento.
- Unidad: representa una unidad u oficina de la organización.
- Usuario: representa al usuario que tiene reservado el documento o que ha firmado el documento.
- Estado: representa el estado en el que se encuentra un documento en un momento dado.
- Definición de metadato: descripción del metadato para ser presentado al usuario.
- Metadato: nombre y valor de los datos de un documento.

El la figura 14 se muestra el modelo de dominio para la biblioteca documental.

ESTRUCTURA DE LA ORGANIZACIÓN

El módulo estructura de la organización define a las unidades que conforman a la organización y sus relaciones de jerarquía, que están dadas por la dependencia de una unidad con otra. Por otro lado, también relaciona a los usuarios con las unidades a las que pertenece y a los usuarios que son jefes de unidad con la unidad que les corresponde supervisar.



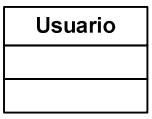
15- Modelo de dominio estructura de la organización

Las restricciones de dominio que se encuentran en este diagrama:

- Un usuario que es jefe de una unidad debe pertenecer a esa unidad.
- Se define una jerarquía entre unidades donde una unidad que depende de otra no puede a la vez supervisarla, como tampoco puede supervisar a ninguna unidad que la supervise.

ADMINISTRACIÓN DE USUARIOS

La administración de usuarios es el módulo que se encarga únicamente del mantenimiento de los usuarios y de su autenticación.



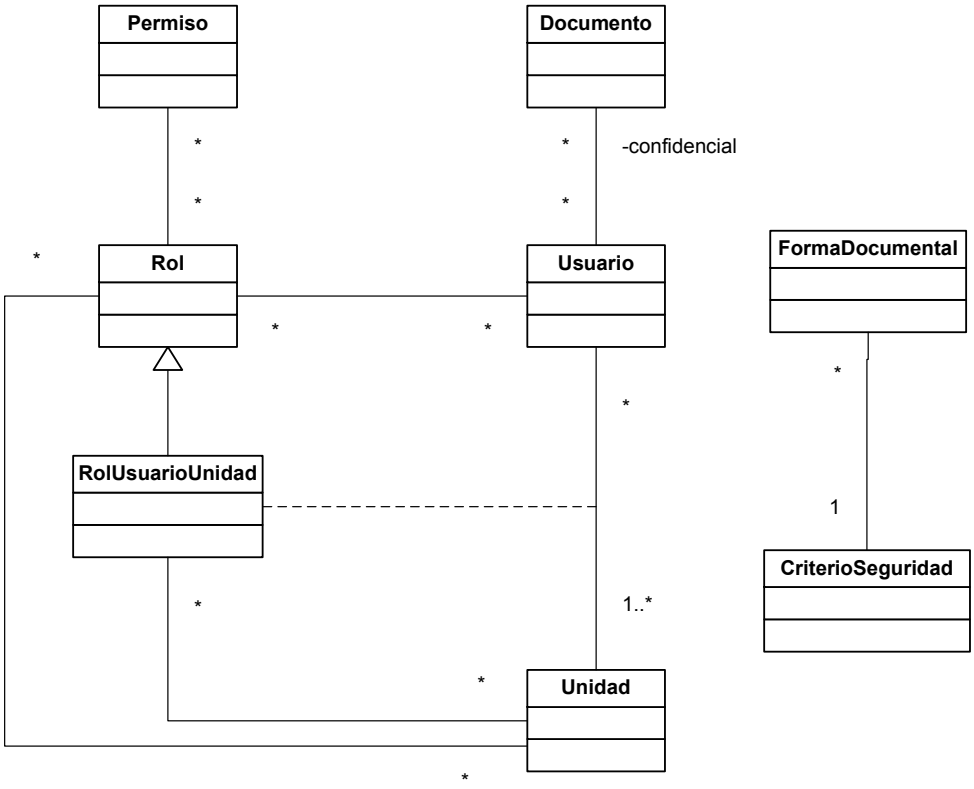
16- Modelo de dominio de la administración de usuarios

SEGURIDAD

El módulo seguridad es el que tiene la responsabilidad del mantenimiento de los permisos de acceso sobre los documentos, las formas documentales y de los permisos de los usuarios, unidades y permisos específicos de usuarios en unidades.

Un rol es una lista de permisos, la asignación de un permiso a un usuario o a una unidad se da a través de un rol que posea ese permiso. Un usuario puede tener determinados permisos en una unidad y solo en ella, esto se diseñó agregando una clase asociación entre usuario y unidad que hereda de la clase rol, por lo tanto los permisos de un usuario en una unidad se definen como permisos de esta clase asociación.

La seguridad de los documentos particulares se indica marcando un trámite como confidencial. Esto se había definido en el documento de requerimientos dentro del módulo de la gestión de los documentos, pero al momento de diseñar se entendió que quien debe tener la responsabilidad de los documentos confidenciales debe ser la seguridad.



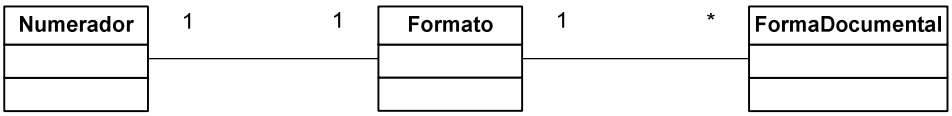
17- Modelo de dominio de seguridad

La seguridad de las formas documentales poseen un criterio de seguridad particular el cual es configurable, dicho criterio puede ser:

- Solo lo pueden ver los usuarios de las oficinas por las que ya paso.
- Son visibles para todos los usuarios sin restricciones a partir de un determinado tiempo o instantáneamente.
- Son visibles para un conjunto de oficinas, luego de un determinado tiempo o instantáneamente.

NUMERADOR

El módulo numerador es el que se encarga de manejar el número de los documentos según el formato que le corresponda. Los distintos tipos de documentos, identificados mediante la forma documental, tienen un formato de numeración que puede ser diferente.



18- Modelo de dominio del numerador

4.3.3. Representación en XML

Uno de los requerimientos funcionales más importantes y en lo que se basa gran parte de la aplicación es en el uso de XML. Esta sección describe el manejo de esos XML.

La biblioteca de documentos se encargará de la gestión de documentos. En la figura 14 se muestra su diagrama de dominio. En este se puede apreciar que los documentos tienen varios metadatos. Los mismos van a ser almacenados en un XML que representa al documento. Cada metadato tiene una definición asociada la que nos va a permitir saber qué tipo de campo es, y como mostrarlo al usuario (por ejemplo: en caso de ser una lista, todos los valores posibles de la misma).

Los documentos van a ser persistidos en una tabla la cual tendrá un atributo de tipo XML, donde se almacena el archivo con los datos.

Los tipos de documento del negocio que son gestionados en la biblioteca son: carátula, actuaciones, secciones, etc. Las formas documentales como expedientes y formularios son una agrupación de estos documentos.

Cada forma documental es la encargada de manejar la agrupación de estos documentos y será la que utilice los servicios ofrecidos por la biblioteca para la creación, mantenimiento, firma, reserva de sus documentos. Además, cada documento tiene asociado una definición que permite presentárselo al usuario con el formato adecuado.

Los módulos que representan cada forma documental (Expediente, Formularios, etc.) se van a encargar de solicitar los documentos pertenecientes a un trámite ya creado, agruparlos en el orden correcto formando el documento XML que representa todo el trámite.

El XML que conforma cada documento tendrá la siguiente estructura:

```
<documento tipo="tipo de documento">
```

```
        <metadato nombre = “nombre del metadato”>
                <valor>
                        Valor en este metadato
                </valor>
                        ……
                <valor>
                        Valor en este metadato
                </valor>
        </metadato>
                ……
        <metadato nombre=“nombre del metadato”>
                ……
        </metadato>
</documento>
```

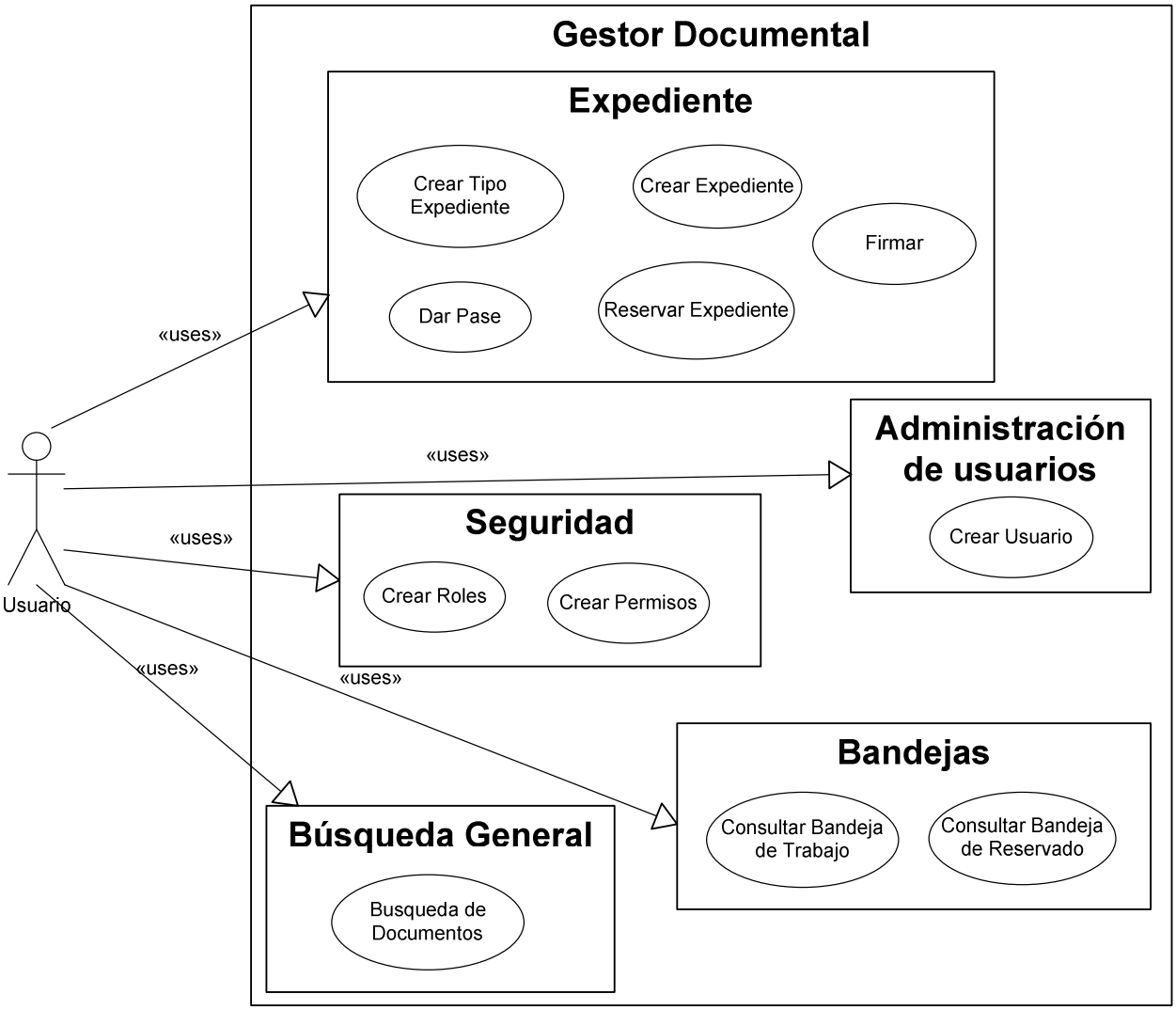
Los XML de las formas documentales van a tener la siguiente estructura:

```
<trámite tipo=“nombre del trámite”>
    <metadato nombre=“nombre del metadato”>
        <valor>
                Valor en este metadato
        </valor>
                ……
        <valor>
                Valor en este metadato
        </valor>
    </metadato>
        ……
    <metadato nombre=“nombre del metadato”>
        ……
    </metadato>
    <documentos>
        Los XML de los documentos que están contenidos en el trámite, respetando
        su orden dentro de la tramitación
    </documentos>
</trámite>
```

4.4. Vista del modelo de caso de uso

Los casos de usos que se identificaron como relevantes a la arquitectura son los siguientes:

- Crear tipo de Expediente
- Crear Expediente
- Dar pase
- Reservar Expediente
- Firmar
- Crear usuario
- Crear roles
- Crear permisos
- Consultar bandeja de trabajo
- Consultar bandeja de reservados
- Búsqueda de documentos



19- Diagramas de casos de uso

Los casos de uso relevantes están especificados en un anexo [Anexo C – C.1] y se diseñan mediante diagramas de secuencia en otro [Anexo C – C.3].

Estos casos de uso son relevantes para la arquitectura dado que cada uno de ellos afecta los diferentes componentes de la solución, de esta forma se tiene una cobertura sustancial de la arquitectura. Por otro lado, también representan las funcionalidades centrales y son significativos para el sistema final.

4.5. Vista del modelo de diseño

4.5.1. Diagrama de arquitectura

Se define una arquitectura de tres capas: capa de presentación, capa lógica y capa de persistencia. Los módulos de seguridad, registro de errores, auditoría y estructura de la organización se comunican con las tres capas, es por eso que se ubicaron como en una capa paralela a lo largo de toda la arquitectura. Esta arquitectura se muestra en la figura 20.

Capa de presentación

Es la capa con la que interactúa el usuario y la que se comunica con la capa lógica para satisfacer las solicitudes del usuario.

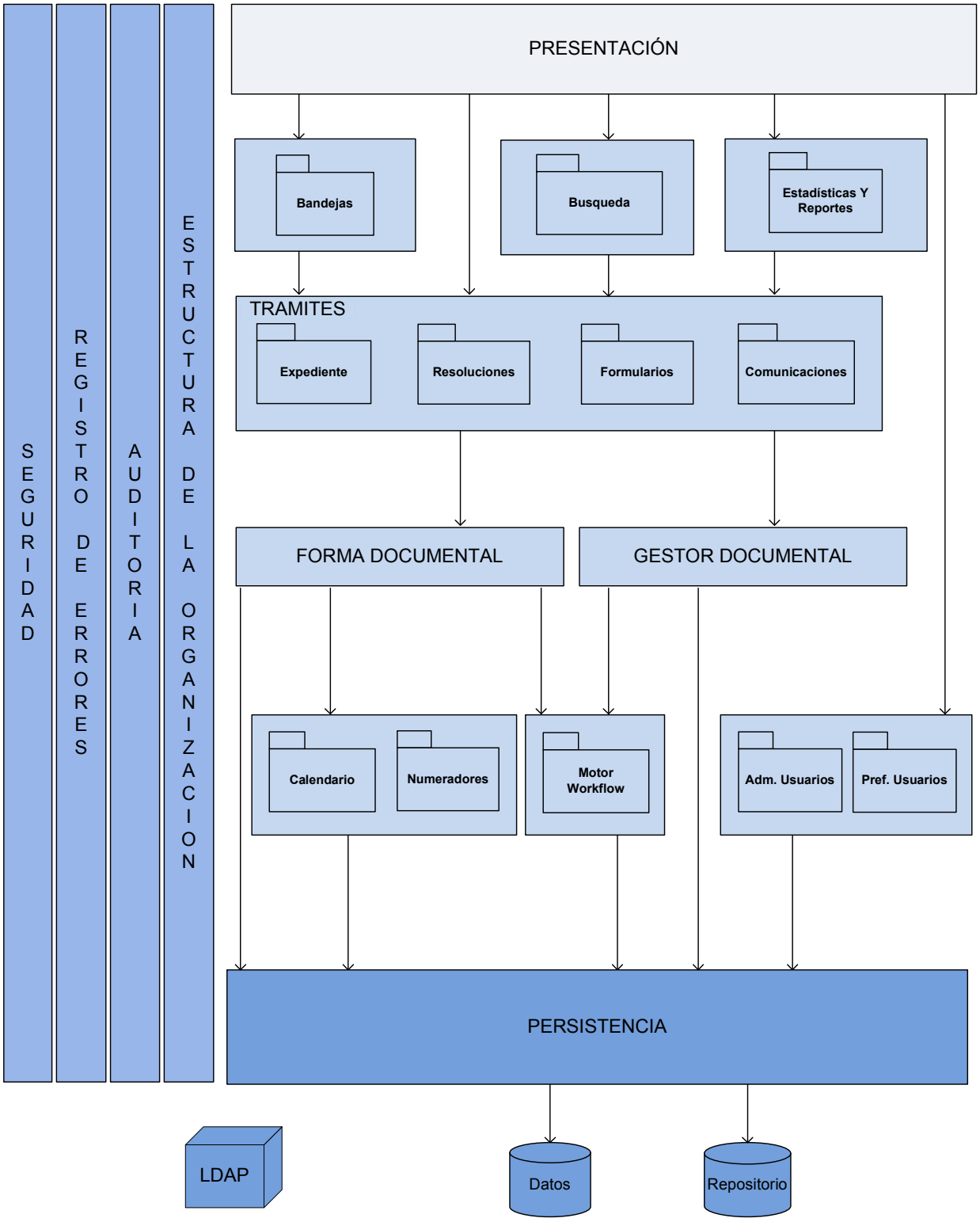
Capa lógica

Esta capa está formada por las entidades, que representan objetos que van a ser manejados o utilizados por toda la aplicación. Dentro de ella se definieron subcapas representando, en el nivel superior las clases con las que se comunica directamente la presentación, un poco más abajo se encuentran los trámites, estos tienen la lógica particular de cada trámite y utilizan los servicios de las formas documentales y el gestor de documentos. Es por esto que dichas librerías aparecen un nivel más abajo debido a que no tienen un acceso directo desde la presentación. Además ofrecen los servicios básicos para el manejo de los trámites y son las encargadas de comunicarse con la capa de persistencia, de esta forma, si se crea un nuevo trámite no es necesario modificar la capa de persistencia. El motor de workflow es utilizado por las librerías para la ejecución de los distintos workflows.

Capa de persistencia

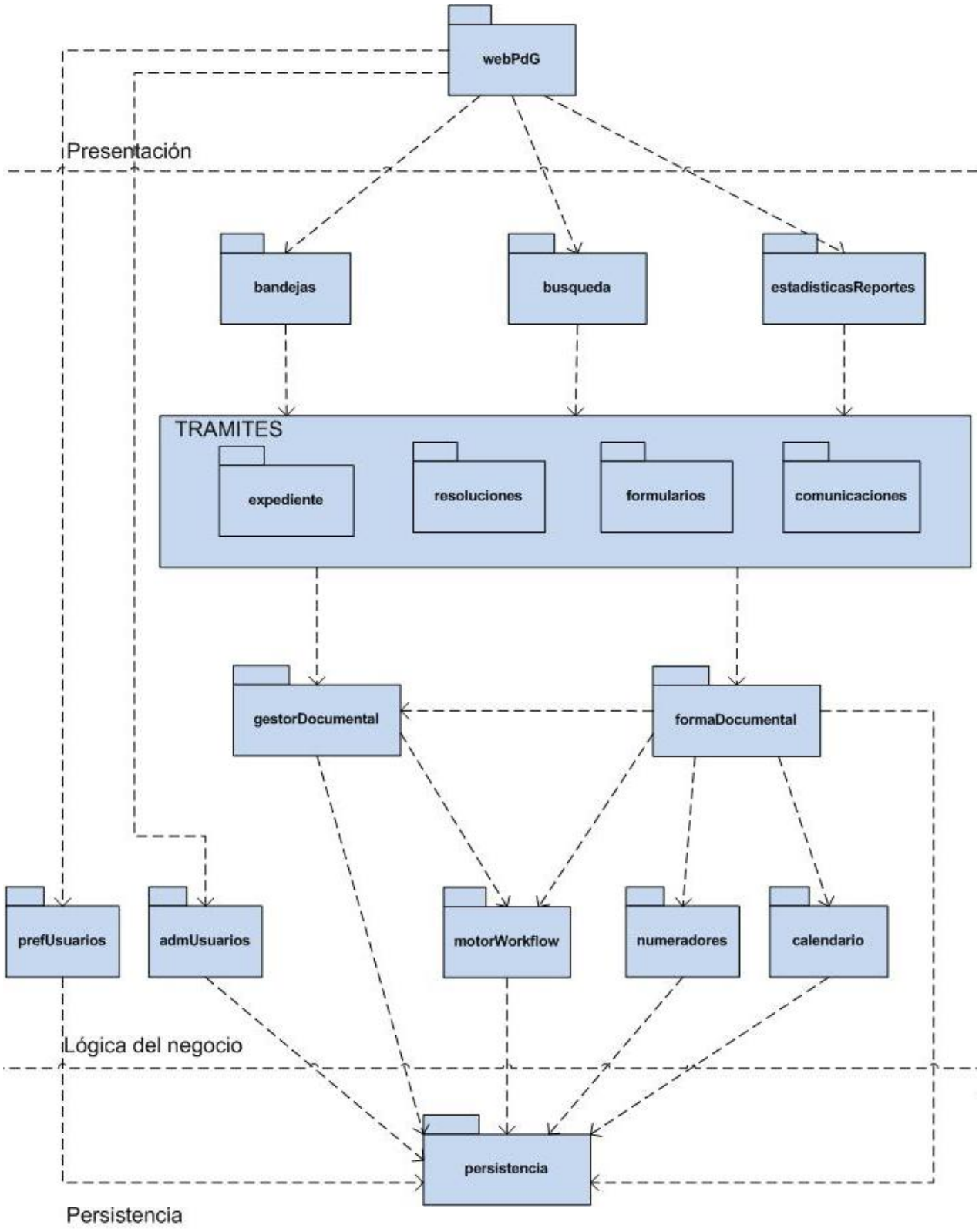
Esta capa contiene clases que interactúan con la base de datos y con el repositorio de archivos adjuntos. Estas clases son especializadas y permiten realizar todas las operaciones con la base de datos y el repositorio de forma transparente para la capa de negocio. Se utilizará en esta capa un framework de persistencia de EJB [77] (Enterprise Java Beans) que utiliza JPA [80] (Java Persistence API).

La comunicación entre capas se realiza mediante el patrón EJBSessions [78] que se describe en la sección 4.5.2.



20- Vista lógica de la arquitectura

DESCOMPOSICIÓN EN SUBSISTEMAS



21- Diagrama de interacción entre subsistemas

Los subsistemas en la arquitectura son:

- *Bandejas*: es el encargado de obtener los documentos para las distintas bandejas que existen en el sistema (Bandeja de Trabajo, Bandeja de Entrada, Bandeja de Salida). Para obtener estos documentos se comunica con las distintas formas documentales, pidiéndoles los documentos para ese usuario en una Unidad.

- *Expediente*: es el encargado de realizar la lógica del trámite Expediente. Para ello va a tener que comunicarse con dos subsistemas que le brindan las funcionalidades necesarias para la creación de sus componentes. Estos subsistemas son: la biblioteca de documentos llamada Gestor Documental y el subsistema Forma Documental, los cuales ofrecen servicios básicos para representar cualquier forma documental.
  El subsistema expediente será el encargado de obtener para un expediente, las definiciones del mismo por medio del subsistema Forma Documental y los documentos que componen ese expediente por medio del subsistema Gestor Documental. La responsabilidad de este subsistema es la implementación de la lógica de negocio del trámite, es decir, la implementación de los requerimientos de ese trámite utilizando los servicios ofrecidos por la capa inferior.

- *Formulario*: este subsistema es similar al subsistema expediente, con la diferencia de que es el encargado de realizar la lógica del tramite formulario.
  Todos los subsistemas de la capa trámites se comportan de igual manera y tienen la misma responsabilidad, realizar la lógica para el trámite que representan utilizando los servicios de la capa inferior en la arquitectura.

- *Forma Documental*: es un subsistema encargado de administrar todas las formas documentales sin conocer su lógica. El subsistema ofrece los servicios básicos a los subsistemas de trámites para que puedan implementar sus requerimientos.
  Este subsistema ofrece funcionalidades que permiten crear un tipo de trámite, crear un trámite de un cierto tipo, poder asignarle definiciones a un tipo de trámite (estas definiciones son creadas por este subsistema sin saber a quién pertenece o para que funcionalidad es; esa lógica es manejada por el subsistema del trámite que invoca la creación de esta definición), poder definir workflow tanto a la forma documental como a un tipo de la misma, ejecutar estos workflows para los trámites creados, etc.
  Toda la lógica representada por cada workflow y por cada definición, es manejada por los subsistemas que consumen estos servicios, los subsistemas de la capa trámites.

- *Gestor Documental*: La biblioteca de documentos está encargada de ofrecer servicios básicos para todos los documentos manejados por el sistema. Este subsistema conoce y administra los documentos sin

diferenciar de que tipo son, ofrece servicios como la creación de un nuevo documento, firmar documentos, puede obtener los documentos contendidos en un trámite, reservar documentos, etc. Los documentos tienen un workflow de estados, permitiendo que cada tipo de documento mantenga su propio workflow.

- *Numeradores*: El subsistema numerador es el que se encarga de asignar un numerador a una forma documental y de otorgar un número a cada nuevo documento de una forma documental dada. También debe poder ser reseteado una vez al año si la configuración así lo establece.

- *Estructura de la organización*: La estructura de la organización es un subsistema que se encarga del mantenimiento de la estructura de la empresa, manejando las relaciones jerárquicas entre las unidades, así como también las relaciones entre los usuarios y las unidades a las que pertenecen.
  Toda la aplicación conoce a la estructura de la organización, por ejemplo, un documento en una unidad es accedido por todos los usuarios de esa unidad, o cuando no se cumple alguno de los tiempos indicados para responder, se notifica utilizando la jerarquía definida.

- *Administración de usuarios*: La administración de usuarios es el subsistema que se encarga del mantenimiento de los usuarios y su autenticación en el sistema.
  Este subsistema interactúa con toda la aplicación ya que solo un usuario autenticado puede acceder a ella.

- *Seguridad*: El subsistema de seguridad es el encargado de mantener a los permisos, a los roles conformados por permisos y a los permisos y roles de cada usuario, unidad y usuario en unidad. Mantiene a los documentos confidenciales y a los usuarios que tienen acceso a esos documentos confidenciales.
  Este subsistema interactúa con toda la aplicación pues cada vez que un usuario intenta acceder a algún documento, obtener información de algo en particular o realizar alguna acción, solo lo podrá realizar si tiene los permisos necesarios y si el documento al cual esta accediendo no es confidencial.

- *Motor de Workflow*: se encarga de la ejecución de los distintos workflows que existen en el sistema. Para esto se utilizan los XPLDs que tienen la definición de los workflows. Además este módulo se encarga de guardar los distintos XPDLs que existen en el sistema.
  Brinda funcionalidades básicas para la ejecución de workflows. Guardando el histórico de las ejecuciones para poder saber el recorrido de las ejecuciones realizadas.

- *Registro de errores*: este módulo se encarga de guardar los errores ocurridos en el sistema.

- *Búsqueda*: este módulo se encarga de la búsqueda de los documentos, según los criterios de búsqueda. Los criterios de búsqueda son los definidos en los requerimientos:
  - Número de documento
  - Fecha de creación
  - Estado en el que se encuentra
  - Texto libre que puede encontrarse en cualquier otro dato del documento
  - Texto libre que puede encontrarse en el contenido del documento y/o archivos anexos a él.

- *Persistencia*: la Persistencia es toda una capa de la arquitectura que se encarga de guardar, modificar y obtener los datos, tanto de la base de datos como del repositorio. Toda la información almacenada debe pasar por esta capa.
  Se crea una clase por cada subsistema de la capa lógica que debe ser almacenado. Dichas clases conocen las tablas donde debe almacenar los datos de ese subsistema. Para el caso de la librería de documentos, va a tener la responsabilidad de manejar el repositorio de anexos de forma transparente para las capas superiores.

Se profundiza sobre los subsistemas mencionados en un anexo [Anexo C – C.2].

4.5.2. Patrones

En esta sección se definirán los patrones que serán aplicados en la solución.

EJBSESSION

Se utilizará el patrón de EJB [77] (Enterprise Java Beans) llamado EJBSession [78] para la comunicación entre las distintas capas. Este patrón permite dar una fachada de acceso a la parte central de la aplicación hecha en EJBs.

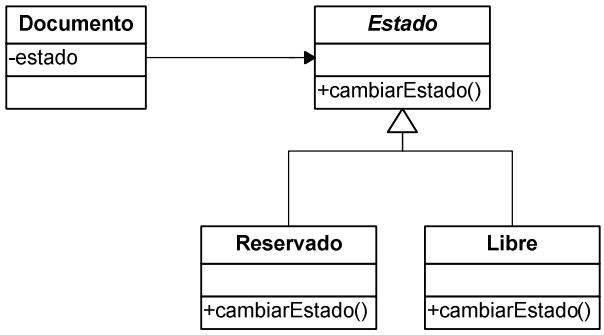
STATE

El patrón State se utiliza para desacoplar la lógica de un objeto de su estado, se mantiene la máquina de estado del objeto de forma independiente del objeto.

El documento conoce que tiene un estado, cuando invoca a la función *cambiarEstado()*, la clase del estado sabe por qué estado debe cambiarse.

Para nuestra realidad los estados existentes son: Reservado y Libre. Al invocar *cambiarEstado()*, el estado Reservado pasa a Libre y el estado Libre pasa a Reservado. Se le agrega una función al estado que tiene la responsabilidad de evaluar si el documento podría ser editado por un

usuario. El estado Reservado retornaría "true" si el usuario es el mismo que tiene reservado el documento. El estado Libre retornaría "true" en todos los casos, indicando que el documento está libre para ser reservado y posteriormente poder ser editado.

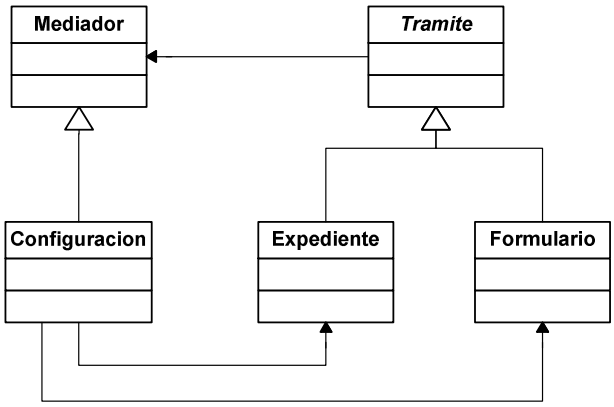


22- Diagrama del Patrón State

MEDIATOR

El patrón Mediador (Mediator), utilizado para la configuración, define un objeto que coordina la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto.

La idea de aplicar este patrón es encapsular el comportamiento de los diferentes trámites en un único objeto. Este objeto conoce que trámites están instalados. Por ejemplo, al momento de realizar una búsqueda se consulta al mediador y no a cada uno de los trámites, es el mediador quien solicita a cada uno de los trámites que posee la información que necesita.

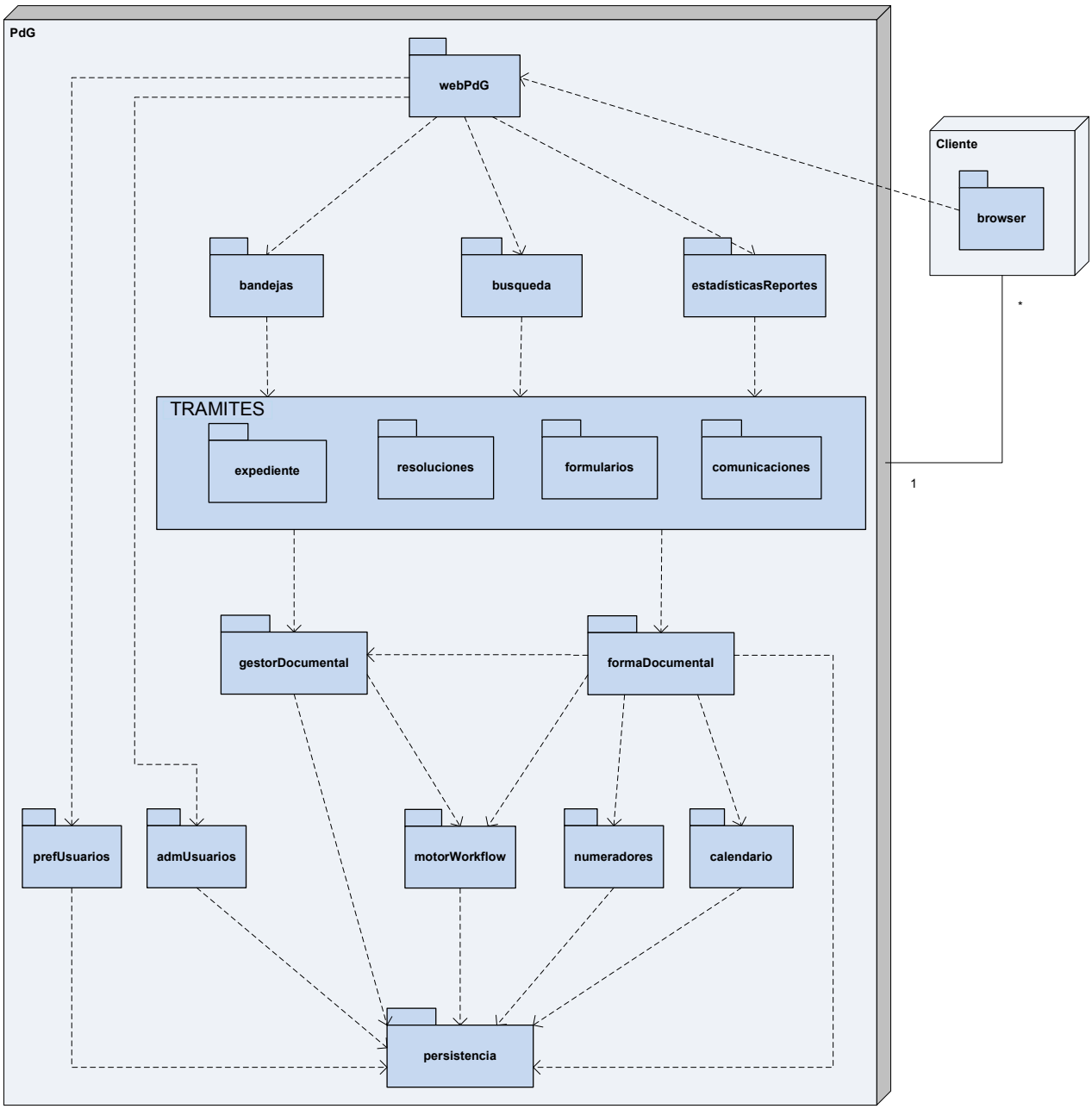


23- Diagrama del Patrón Mediator

4.6. Vista del modelo de distribución

4.6.1. Diagramas de distribución

El diagrama de distribución de la figura 24 muestra una posible distribución física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional. Los nodos se utilizan para modelar la topología del hardware sobre el que se ejecuta el sistema.



24- Diagrama de distribución

4.6.2. Nodos

CLIENTE

Este nodo representa la máquina cliente que desea enviar solicitudes vía Web al sistema, es necesario poder utilizar un web browser desde la misma.

SERVIDOR

En este nodo se encuentra toda la lógica de la aplicación, así como las bases de datos. Por lo tanto, necesita tener un motor de bases de datos instalado así como un servidor web. Se necesita una máquina de buena capacidad y con suficiente memoria para soportar las tecnologías necesarias y para tener un buen desempeño.

Este nodo es el que brinda los servicios requeridos por el nodo cliente.

Además, debe tener conexión a Internet para poder conectarse con otras bases de datos en el mundo y poder bajar información requerida.

4.6.3. Conexiones

CLIENTE – SERVIDOR

Se necesita una conexión que soporte http.

Capítulo 5 Prototipo

5.1. Introducción

Entre los objetivos del proyecto se encuentra el de desarrollar un prototipo que utilice las tecnologías investigadas y seleccionadas como las mejores opciones en la etapa de análisis de tecnología, demostrando que la solución planteada en el diseño es una solución factible y que la combinación de tecnologías seleccionadas es posible.

En este capítulo se describe el prototipo desarrollado, las tecnologías utilizadas y el alcance del mismo.

5.2. Metodología de trabajo

Se realizó una estimación de los requerimientos y en base a ella, junto al cliente, se definió el alcance final del prototipo.

Los módulos dentro del alcance final para el prototipo son:

- Estructura: con las funcionalidades básicas que permita el funcionamiento de las formas documentales.
- Administración de usuarios: con las funcionalidades simples de registro de usuarios.
- Bandejas: implementando las tres bandejas principales (de entrada, de trabajo y de salida).
- Búsqueda: permitiendo búsqueda full-text tanto sobre los documentos que integran cada trámite como sobre los archivos adjuntos que estos puedan tener.
- Forma Documental Expediente.
- Motor de Workflow.
- Módulo de Forma Documental.
- Biblioteca de Documentos.

Quedando fuera del alcance, la seguridad del sistema, las restantes formas documentales y funcionalidades como firma digital, numeración automática, notificaciones, calendario.

Se realizaron dos entregas al cliente durante la etapa de desarrollo, las cuales consistieron en la presentación de las funcionalidades desarrolladas mostrando el avance. De estas reuniones se obtuvieron sugerencias para mejoras y aprobación del resultado obtenido, teniendo en cuenta el alcance del prototipo.

La primer entrega contempló las funcionalidades de las bibliotecas de documentos, administración de usuarios, estructura y la forma documental Expediente. Para la segunda entrega se incorporaron las funcionalidades

faltantes contempladas por el alcance, tales como, el diseño de la ruta de forma gráfica, la búsqueda full-text, y mejoras sugeridas por el cliente en la entrega anterior. Dada la flexibilidad del diseño para la incorporación de una nueva forma documental y para demostrar que realmente es así, se decidió agregar la Forma Documental Formulario, aunque esto no estaba previsto en el alcance.

5.3. Implementación

Para la implementación del prototipo se utilizaron las tecnologías elegidas en la etapa de análisis de estas.

A continuación se detallan en resumen estas tecnologías:

- *J2EE* [27] en su versión 1.6.
- JBoss [79] versión 4.2 como servidor de aplicaciones web.
- *Session Beans* (*Stateless*, "sin estado" en español) de *EJB3* [77] para la comunicación entre las distintas capas de la arquitectura.
- *JPA* [80] para la persistencia de los datos. Esta API permite acelerar los tiempos de implementación realizando la creación y actualización de las tablas del sistema en forma muy simple, así como también la persistencia y obtención de los datos. Además se encarga del manejo de las transacciones facilitando al implementador.
- *JDom* [81] como librería java para el manejo de XMLs.
- Para la edición de las rutas que pueden seguir los distintos documentos que integran los trámites, se utilizó la librería JavaScript [18] llamada *MxGraph* [16]. Esto nos permitió integrar en nuestra aplicación web un diseñador de rutas gráfico. Para llevar a cabo este cometido, se adaptó el código JavaScript de esta librería permitiendo hacer el traspaso de información de la aplicación a esta librería y viceversa.
- Como herramienta para la creación de las páginas webs se utilizó *ZK* [31] versión 3.5.2 creando una rica interfaz en poco tiempo y permitiendo la integración con *MxGraph*.
- Como IDE se utilizó *Eclipse Europa* versión 3.3.2 [33] con los plugins necesarios para el manejo de *JBoss*, *JPA*, *EJB3*, *ZK*, etc.
- *SVN* (Subversion Native Library Adapter 1.5.3) [82] como sistema de control de versiones, permitiendo la sincronización de los cambios en el código, realizados por los distintos integrantes del grupo, de una manera muy sencilla y eficaz.
- Como Base de Datos se utilizó *Postgres* [39] con las extensiones para el soporte de XML y de la búsqueda full-text (*tsearch2* [76]).
- *Jackrabbit* [30] como repositorio de archivos, para el manejo de los archivos adjuntos permitiendo realizar búsquedas full-text dentro de estos archivos.

FICHA TÉCNICA

A continuación se detallan las configuraciones y proyectos de desarrollo utilizados para la implementación de la solución del prototipo.

Se tienen cuatro proyectos en el IDE (Eclipse):

- *pdg_logic* – contiene la lógica de la aplicación.
  - En la carpeta META-INF se le configura el data-source a utilizar.
  - Está formada por una estructura de paquetes que respetan a los subsistemas definidos en la arquitectura.
- *pRoyecto* – contiene la aplicación web con las librerías de ZK.
- pdg*_repositorio* – se encarga de la comunicación con el repositorio de archivos (Jackrabbit) brindando las funcionalidades necesarias para poder agregar, obtener y buscar archivos en el repositorio.
- *pdg_jackrabbit* – controla el acceso al repositorio, contiene una clase que es la encargada de controlar el acceso de los distintos usuarios a los archivos. En este prototipo no hay restricciones de acceso a los usuarios.

Cada proyecto tiene un archivo llamado "build.xml", que ejecuta comandos (Ant) que generan los archivos '.jar' o '.war' correspondientes y realizan el deploy de estos en el JBoss.

Además en el JBoss se tienen los siguientes archivos:

- *postgre-ds.xml* – contiene la conexión JDBC [83] con la base de datos Postgres.
- *jcr-ds.xml* – contiene la información para la creación inicial y la conexión con el repositorio de archivos. Además posee la ruta al Sistema de Archivos, en donde quedarán los archivos y la configuración del repositorio.
- *jackrabbit-jca-1.4.rar* – contiene las librerías necesarias para la conexión con el repositorio de archivos.
  - Aquí se agrega el correspondiente archivo '.jar' generado por el proyecto *pdg_jackrabbit* llamado *PdGAccessManagerForJackRabbit.jar* y se configura en *repository.xml* para que tome la clase que se encuentra en ese archivo para manejar el acceso al repositorio.

5.4. Funcionalidades

Se implementaron las funcionalidades principales en el sistema de Gestión Documental, respetando los requerimientos funcionales y no funcionales, además del diseño flexible para la incorporación de nuevas formas documentales.

En resumen, las funcionalidades que se ofrecen en el prototipo son:

- El mantenimiento de la estructura, definiendo usuarios, jefes y unidades.

- La biblioteca básica de documentos Gestor Documental, ofrece la búsqueda sobre los documentos, la definición de un diagrama de estados para los distintos tipos de documentos y el manejo de archivos adjuntos.

- El módulo de Expedientes, que incluye la creación, modificación de un expediente, las funcionalidades del flujo como *dar pase*, *archivar*, *reservar*, *liberar*, etc.

- Para Formularios, se ofrecen las mismas funcionalidades que para el módulo Expedientes, con las diferencias de negocio que tiene con respecto al mismo.

- En el módulo de Forma Documental, se ofrecen las funcionalidades que son utilizadas por expedientes y formularios, tales como la creación y modificación de una Forma Documental. Permite la definición de tipos de formas documentales a las cuales se les pueden incorporar workflows de forma gráfica utilizando un editor gráfico integrado con las otras funcionalidades de la aplicación y definiéndoles los datos que se interesa cargar para ese tipo de trámite.

- Se muestran las diferentes bandejas seleccionando una unidad, en cada bandeja los trámites que posee junto con la lista de acciones que se pueden ejecutar sobre cada uno.

Capítulo 6 Resultados Obtenidos

Dentro de los principales resultados obtenidos durante el transcurso del proyecto se destaca la obtención de un buen nivel de especificación de requerimientos. Esto se debe al tiempo dedicado a esta actividad, junto con el gran conocimiento que posee el cliente. Dado que el cliente desde hace varios años conoce el negocio y trabaja con distintas organizaciones, se pudieron obtener las necesidades generales y las particularidades que debe ofrecer un sistema de Gestión Documental para ser adaptable a las diferentes organizaciones.

Como resultado de la etapa de análisis de tecnologías se obtuvo un resumen de las tecnologías que se podrían llegar a utilizar para la realización de un sistema de Gestión Documental, planteando diferentes alternativas sobre las principales decisiones que se deben tomar al momento de implementar un sistema de este tipo.

En el diseño se generó una definición de la arquitectura del sistema completo, basándose en el relevamiento de requerimientos y previendo cambios en el negocio que pudieren surgir. Se llegó a un diseño de bajo nivel sobre los casos de uso que definen la arquitectura, realizando por ejemplo, diagramas de actividad.

En la implementación del prototipo se pudo evaluar tanto la elección de las tecnologías como el diseño realizado.

Entre las decisiones tomadas en la etapa de análisis de tecnologías, se decidió utilizar una base de datos relacional con soporte XML, lo cual facilitó la implementación del prototipo (utilizando frameworks para la persistencia java y el conocimiento ya adquirido sobre bases de datos relacionales) y permitió manejar la estructura XML de los documentos sin necesidad de una transformación a datos relacionales, análogamente, no se requirió manejar los módulos relacionales en una base de datos nativa XML.

Para cumplir con los requerimientos de almacenamiento y búsqueda en los adjuntos, dado que en la investigación no se encontró una alternativa que cumpliera con esos requerimientos utilizando una única fuente de almacenamiento que fuese open-source, se utilizó un repositorio de archivos separado a la base de datos, logrando con esto el cumplimiento de ambos requerimientos sin mayores complicaciones en el manejo de ambas fuentes.

Otra decisión importante tomada durante la investigación, fue la utilización de un editor gráfico para la definición de rutas y diagramas de estado. Respecto a esta decisión, se logró cumplir el requerimiento de diseñar gráficamente la ruta de unidades de forma intuitiva e integrada en la aplicación. No se realizó la definición de diagramas de estado, dado que

quedó fuera del alcance para el prototipo, pero se probó que es una buena solución.

Para la ejecución de los workflows se implementaron las necesidades básicas de un motor, que permitió el cumplimiento de los requerimientos. Como decisión para el prototipo se considera que fue una buena alternativa dado el alcance del mismo, pero la implementación realizada no es lo suficientemente eficiente como para un desarrollo mayor. Dado que se utilizó un estándar se podría cambiar la implementación del motor por un motor eficiente y con mayor cantidad de funcionalidades.

En la implementación se obtuvo un prototipo que integró las tecnologías elegidas cumpliendo con el diseño del sistema. El alcance del prototipo fue limitado pero suficiente para desarrollar los principales requerimientos de un sistema de Gestión Documental.

Capítulo 7 Conclusiones y Futuros Trabajos

Se cumplió con los objetivos planteados al principio del proyecto. Dentro de las principales etapas se encuentran: el relevamiento de requerimientos, la investigación de tecnologías y el diseño del sistema. Estas etapas fueron las más importantes durante el proyecto y a las que se le dedicó la mayor parte del tiempo. Se lograron buenos resultados y la satisfacción del cliente respecto a los mismos.

También se cumplió con la realización de un prototipo que cumpliera con los objetivos planteados por el cliente: una aplicación independiente del navegador Web, que ofrece búsqueda full-text en documentos y adjuntos, utilizando tecnologías open-source, implementado en java, independiente del sistema operativo, etc.

El proyecto se realizó de forma adecuada, cumpliendo con la planificación del mismo y las entregas para cada una de las etapas en tiempo y forma.

Se pueden mencionar como futuros trabajos en la continuidad de este proyecto, implementar las funcionalidades faltantes en el prototipo. Entre las cuales se destacan:

- Incorporación de la seguridad y control de roles de los usuarios.
- Aprovechar las funcionalidades de búsqueda que ofrecen las herramientas utilizadas en el almacenamiento de datos, para remarcar la información del texto donde se encuentran las palabras buscadas.
- Mejorar la búsqueda full-text en Postgres [39], utilizando los diccionarios de tsearch2, para que se encuentren palabras similares a las que se quieren buscar.
- Realizar un buen diseño de interfaz de usuario, mejorando la usabilidad del sistema.
- Incorporar nuevos tipos de metadatos de los documentos como puede ser, poder definir un campo de tipo fecha. Además agregar la posibilidad de indicar que campos son requeridos en el documento.
- Agregar el diseño gráfico de los diagramas de estado, al igual que el implementado para la definición de rutas de unidades.
- Mejorar la implementación del motor de workflows, haciéndolo más eficiente e incorporando más funcionalidades.
- Incorporarle a la definición de los documentos, que además de elegir los campos que el usuario va a ingresar para el mismo, también pueda diseñar la presentación de estos documentos, tanto para su visualización como para su modificación.

- Incorporar nuevas Formas Documentales de la misma manera que se incorporó al alcance del prototipo la forma Formularios. Las Formas Documentales ya relevadas que se podrían incorporar son Resoluciones y Comunicaciones.

Glosario

Ad-hoc: se refiere a una solución elaborada específicamente para un problema o fin preciso y, por tanto, no es generalizable ni utilizable para otros propósitos. Se usa para referirse a algo que es adecuado sólo para un determinado fin.

AJAX: acrónimo para Asynchronous JavaScript And XML, es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

BPEL: Business Process Execution Language, WS-BPEL (Lenguaje de Ejecución de Procesos de Negocio), es un lenguaje estandarizado por OASIS para la composición de servicios web. Está desarrollo a partir de WSDL y XLANG, ambos lenguajes orientados a la descripción de servicios Web. Básicamente, consiste en un lenguaje basado en XML diseñado para el control centralizado de la invocación de diferentes servicios Web, con cierta lógica de negocio añadida que ayuda a la programación en gran escala (programming in the large).

BPM: Business Process Management, metodología empresarial cuyo objetivo es mejorar la eficiencia a través de la gestión sistemática de los procesos de negocio, que se deben modelar, automatizar, integrar, monitorizar y optimizar de forma continua.

BPMN: Business Process Management Notation (Notación para el Modelado de Procesos de Negocio), es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo (workflow).

Comunicación: trámites que se utilizan para la comunicación entre personas u oficinas de la organización.

Decreto 500/991: decreto de la legislación uruguaya donde se especifican algunos de los trámites y formas de trabajo sobre ellos. Las disposiciones de este decreto alcanzan al procedimiento administrativo común, desenvuelto en la actividad de los órganos de la Administración Central y a los especiales técnicos en cuanto condigan con su naturaleza.

Definición de Metadato: es la información que define a un atributo, contiene el nombre con el que se identifica, el tipo de dato que se le asigna, si es requerido o no, lista de valores o rangos posibles, etc.

E-government: consiste en el uso de las tecnologías de la información y el conocimiento en los procesos internos de gobierno y en la entrega de los productos y servicios del Estado tanto a los ciudadanos como a la industria. Muchas de las tecnologías involucradas y sus implementaciones son las mismas o similares a aquéllas correspondientes al sector privado del comercio electrónico (o e-business), mientras que otras son específicas o únicas en relación a las necesidades del gobierno.

EJB: Enterprise Java Beans, es una de las API que forman parte del estándar de construcción de aplicaciones empresariales J2EE de Sun Microsystems. Los EJB proporcionan un modelo de componentes distribuido estándar del lado del servidor. El objetivo de los EJB es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, etc.) para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes permite que éstos sean flexibles y sobre todo reutilizables.

Expediente: trámite que contiene una serie de actuaciones (documentos) realizadas por la organización sobre el asunto.

Flujo de trabajo: workflow, es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas.

Formas Documentales: término utilizado para llamar de forma genérica a los módulos de Expedientes, Formularios, Resoluciones, Comunicaciones, etc.

Formulario: trámite administrativo estructurado que contiene una lista ordenada de secciones con información específica a completar en cada etapa.

Hibernate: herramienta de software libre de Mapeo objeto-relacional para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

J2EE (Java 2 Enterprise Edition): edición empresarial del paquete Java creada y distribuida por Sun Microsystems. Comprenden un conjunto de especificaciones y funcionalidades orientadas al desarrollo de aplicaciones empresariales.

JBoss: es un servidor de aplicaciones J2EE de código abierto implementado en Java puro.

jBPM: Gestor de procesos de negocio de JBoss, también denominado "WorkFlow".

JPA: Java Persistence Api, API de persistencia desarrollada para la plataforma JavaEE e incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como sí pasaba con EJB2, y permitir usar objetos regulares.

JSF: Java Server Faces, es un framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE. JSF usa JavaServer Pages (JSP) como la tecnología que permite hacer el despliegue de las páginas, pero también se puede acomodar a otras tecnologías.

JSP: Java Server Pages, es una tecnología Java desarrollada por Sun Mycrosystems, que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.

Metadato: es un dato de un documento formado por el nombre del campo y su valor o valores.

Model View Controller: patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Motor de workflow: es una herramienta que permite dar forma y automatizar los procesos de negocios de la empresa. Con este tipo de herramientas se pueden formalizar las reglas comerciales de la empresa para automatizar el proceso de toma de decisiones.

Resolución: trámites que representa la decisión tomada en una reunión por el directorio o la alta esfera jerárquica de una organización.

RMI: (Java Remote Method Invocation) es un mecanismo ofrecido en Java para invocar un método remotamente.

Seam Pageflow: definición de proceso o flujo de negocio. Para su representación se utiliza jPDL que tiene sintaxis XML y puede editarse gráficamente con un plugin de eclipse. En dicha representación se concretan los diferentes estados, decisiones, tareas, páginas web, etc. con las cuales interactúa el usuario.

Spring: framework de código abierto de desarrollo de aplicaciones para Java.

Struts: herramienta de código abierto de soporte para el desarrollo de aplicaciones Web bajo el patrón MVC bajo la plataforma J2EE (Java 2, Enterprise Edition). Se desarrollaba como parte del proyecto Jakarta de

la Apache Software Foundation, pero actualmente es un proyecto independiente conocido como Apache Struts.

Trámite: proceso administrativo por el que tiene que pasar un asunto para ser solucionado.

Web-services: colección de protocolos y estándares que sirve para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes y ejecutadas sobre cualquier plataforma pueden utilizar los web-services para intercambiar datos.

XML: Extensible Markup Language («lenguaje de marcas ampliable»), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML).

XPDL: (XML Process Definition Language) es un lenguaje para la definición de un Flujo de trabajo.

Referencias

[1] Sitio de Presidencia donde se encuentra la legislación uruguaya, los decretos. http://www.presidencia.gub.uy/
Última consulta: marzo 2009

[2] Sitio del Parlamento del Uruguay, donde se encuentran las leyes.
http://www.parlamento.gub.uy/palacio3/index1024.htm
Última consulta: marzo 2009

[3] Sitio oficial de Urudata. http://www.urudata.com/
Última consulta: marzo 2009

[4] Sitio oficial de Arson Group, que posee la herramienta QExpeditive.
http://www.arsongroup.com/
Última consulta: marzo 2009

[5] Sitio oficial de la empresa ISA Ltda. http://www.isaltda.com.uy
Última consulta: marzo 2009

[6] Sitio oficial de Integradoc.
http://www.integradoc.com/integradoc/index.html
Última consulta: marzo 2009

[7] Modelo de requisitos para la gestión de documentos electrónicos de archivo: Especificación MoReq.
http://www.csae.map.es/csi/pg5m52.htm
Última consulta: marzo 2009

[8] Sitio oficial de Gedex, Gestión de expedientes jurídicos para despacho de abogados. http://www.brindys.com/gedex/
Última consulta: marzo 2009

[9] Sitio oficial de asesoweb, empresa que implementa sdmEXP.
http://www.asesoweb.com/index2.html
Última consulta: marzo 2009

[10] Sitio oficial de AEAT, Agencia Tributaria de España.
http://www.aeat.es
Última consulta: marzo 2009

[11] Wikipedia. http://es.wikipedia.org/wiki/Wikipedia:Portada
Última consulta: marzo 2009

[12] Sitio oficial de GCIC, gestión y custodia de la información.
http://www.gcic.es/gcic/GCIC/published/node_234.html
Última consulta: marzo 2009

[13] Sitio oficial de Expe+. http://www.expe.edu.uy/

Última consulta: marzo 2009

[14] Sitio oficial de AGESIC, Agencia para el desarrollo del gobierno de gestión electrónica y la sociedad de la información y del conocimiento. http://www.agesic.gub.uy/Sitio/default.asp
Última consulta: marzo 2009

[15] Sitio oficial de Together, donde se encuentra el Together Workflow Editor y Enhydra Shark.
http://www.together.at/together/index.php
Última consulta: marzo 2009

[16] Sitio oficial de MxGraph. http://www.mxgraph.com/index.html
Última consulta: marzo 2009

[17] Sitio oficial de GEF Tigris. http://gef.tigris.org/
Última consulta: marzo 2009

[18] Especificación oficial de Javascript.
http://www.w3schools.com/JS/default.asp
Última consulta: marzo 2009

[19] Sitio oficial de Java. http://www.java.com/
Última consulta: marzo 2009

[20] Sitio oficial de la base de datos DB2, IBM. http://www-01.ibm.com/software/data/db2/
Última consulta: marzo 2009

[21] Sitio oficial de la base de datos HSQL. http://hsqldb.org/
Última consulta: marzo 2009

[22] Sitio oficial de MySQL. http://www.mysql.com/
Última consulta: marzo 2009

[23] Sitio oficial de Oracle. http://www.oracle.com/
Última consulta: marzo 2009

[24] Sitio de Microsoft de SQLXML. http://msdn.microsoft.com/en-us/library/aa286527.aspx#
Última consulta: marzo 2009

[25] Sitio oficial de WfMC, Workflow Management Coalition.
http://wfmc.org/
Última consulta: marzo 2009

[26] Sitio oficial de la base de datos Intalio.
http://bpms.intalio.com/
Última consulta: marzo 2009

[27] Sitio oficial de J2EE dentro del sitio de Java. http://java.sun.com/javaee/
Última consulta: marzo 2009

[28] Especificación oficial de XML. http://www.w3.org/XML/
Última consulta: marzo 2009

[29] Especificación de XPDL. http://www.wfmc.org/XPDL/
Última consulta: marzo 2009

[30] Sitio oficial de Apache Jackrabbit. http://jackrabbit.apache.org/
Última consulta: marzo 2009

[31] Sitio oficial de ZK. http://www.zkoss.org/
Última consulta: marzo 2009

[32] Sitio oficial de Java Beans en Java. http://java.sun.com/javase/technologies/desktop/javabeans/index.jsp
Última consulta: marzo 2009

[33] Sitio oficial de Eclipse. http://www.eclipse.org/
Última consulta: marzo 2009

[34] Documentación sobre WS-BPEL Extension for People, BPEL4People. http://www.ibm.com/developerworks/webservices/library/specification/ws-bpel4people/
Última consulta: marzo 2009

[35] Sitio de las especificaciones JSR (Java Specification Request), JSR 168. http://jcp.org/en/jsr/detail?id=168
Última consulta: marzo 2009

[36] Especificación de XForms. http://www.w3.org/MarkUp/Forms/
Última consulta: marzo 2009

[37] Sitio oficial de JBoss jBPM. https://www.jboss.com/products/jbpm/
Última consulta: marzo 2009

[38] Sitio oficial de Microsoft SQL Server. http://www.microsoft.com/spain/sql/default.mspx
Última consulta: marzo 2009

[39] Sitio oficial de PostgreSQL. http://www.postgresql.org/
Última consulta: marzo 2009

[40] Sitio oficial de XML:DB Initiative. http://xmldb-org.sourceforge.net/

Última consulta: marzo 2009

[41] Informatizate, paper sobre bases de datos nativas XML. http://www.informatizate.net/articulos/bases_de_datos_nativas_en_xml_20020712.html
Última consulta: marzo 2009

[42] Especificación de XPath. http://www.w3.org/TR/xpath
Última consulta: marzo 2009

[43] Especificación de XML Infoset. http://www.w3.org/TR/xml-infoset/
Última consulta: marzo 2009

[44] Especificación de DOM. http://www.w3.org/DOM/
Última consulta: marzo 2009

[45] Sitio oficial de SAX. http://www.saxproject.org/
Última consulta: marzo 2009

[46] Especificación de XQuery. http://www.w3.org/TR/xquery/
Última consulta: marzo 2009

[47] Sitio de la base de datos de Ozone/XML. http://www.ozone-db.org/noframes/ozonies/ozonexml.html
Última consulta: marzo 2009

[48] Artículo sobre Full-Text Search en SQL Server. http://www.webtaller.com/construccion/lenguajes/sql/lecciones/full-text-search-sql-server.php
Última consulta: marzo 2009

[49] Especificación de SQL. http://www.w3schools.com/sql/default.asp
Última consulta: marzo 2009

[50] Sitio oficial de la base de datos Textml. http://www.ixiasoft.com/default.asp
Última consulta: marzo 2009

[51] Artículo "Introducing the Java Content Repository API". http://www.ibm.com/developerworks/java/library/j-jcr/
Última consulta: marzo 2009

[52] Sitio oficial de Apache Software Foundation. http://www.apache.org/
Última consulta: marzo 2009

[53] Sitio oficial de Lucene. http://lucene.apache.org/java/docs/
Última consulta: marzo 2009

[54] Sitio oficial del W3C, World Wide Web. http://www.w3c.es/
Última consulta: marzo 2009

[55] Sitio oficial de JDom. http://www.jdom.org/
Última consulta: marzo 2009

[56] Sitio oficial de NetBeans. http://www.netbeans.org/
Última consulta: marzo 2009

[57] Especificación de CSS. http://www.w3.org/Style/CSS/
Última consulta: marzo 2009

[58] Especificación de RDF. http://www.w3.org/RDF/
Última consulta: marzo 2009

[59] Artículo "BBDD Nativas y sistemas de recuperación HTML, RDF, XML". http://www.geocities.com/basesdatosnativas/bdnativas.html
Última consulta: marzo 2009

[60] Especificación de XHTML. http://www.w3.org/TR/xhtml1/
Última consulta: marzo 2009

[61] Especificación de XML Schema.
http://www.w3.org/XML/Schema
Última consulta: marzo 2009

[62] Especificación de XSLT. http://www.w3.org/TR/xslt
Última consulta: marzo 2009

[63] Sitio oficial de Open Symphony OSWorkflow.
http://www.opensymphony.com/osworkflow/
Última consulta: marzo 2009

[64] Sitio oficial de WfMOpen. http://wfmopen.sourceforge.net/
Última consulta: marzo 2009

[65] Sitio oficial de Bonita.
http://wiki.bonita.objectweb.org/xwiki/bin/view/Main/WebHome
Última consulta: marzo 2009

[66] Sitio oficial de OpenWFE. http://www.openwfe.org/index.html
Última consulta: marzo 2009

[67] Sitio oficial de Zebra. http://zebra.berlios.de/
Última consulta: marzo 2009

[68] Sitio oficial de Yawl. http://yawlfoundation.org/
Última consulta: marzo 2009

[69] Sitio oficial de Imixs. http://www.imixs.org/
Última consulta: marzo 2009

[70] Sitio oficial de XFlow. http://xflow.sourceforge.net/
Última consulta: marzo 2009

[71] Especificación de la base de datos Tamino. http://www.softwareag.com/Corporate/products/wm/tamino/default.asp
Última consulta: marzo 2009

[72] Sitio oficial de Sedna. http://modis.ispras.ru/sedna/index.htm
Última consulta: marzo 2009

[73] Sitio oficial de Exist. http://exist.sourceforge.net/
Última consulta: marzo 2009

[74] Sitio oficial de IBM Lotus Notes y Domino. http://www-01.ibm.com/software/lotus/notesanddomino/
Última consulta: marzo 2009

[75] Sitio oficial de IBM DB2 Express-C - http://www-01.ibm.com/software/data/db2/express/download.html
Última consulta: marzo 2009

[76] Documentación sobre tsearch2 – full-text extension for postgreSQL.
http://www.sai.msu.su/~megera/postgres/gist/tsearch/V2/
Última consulta: marzo 2009

[77] Sitio oficial de EJB dentro del sitio de Java - http://java.sun.com/products/ejb/
Última consulta:

[78] Patrón EJB Session - http://java.sun.com/blueprints/corej2eepatterns/Patterns/SessionFacade.html
Última consulta: marzo 2009

[79] Sitio oficial de JBoss dentro del sitio de Java - http://java.sun.com/blueprints/corej2eepatterns/Patterns/SessionFacade.html
Última consulta: marzo 2009

[80] Java Persistence Api (JPA) - http://java.sun.com/developer/technicalArticles/J2EE/jpa/
Última consulta: marzo 2009

[81] JDOM (Java Document Object Model) – Biblioteca para manejo de XML en java - http://www.jdom.org/
Última consulta: marzo 2009

[82] Sitio oficial de SubVersion (SVN) -
http://es.wikipedia.org/wiki/Subversion
Última consulta: marzo 2009

[83] Sitio oficial de JDBC dentro del sitio de Java -
http://java.sun.com/products/jdbc/overview.html
Última consulta: marzo 2009

[84] Sitio oficial de Domus -
http://www.arnaldocastro.com.uy/domus/
Última consulta: marzo 2009

[85] Sitio oficial de VERS (Victorian Electronics Records Strategy)
http://www.prov.vic.gov.au/vers/vers/default.htm
Última consulta: marzo 2009

[86] Sitio de Patrones de Workflow de YAWL -
http://www.workflowpatterns.com
Última consulta: marzo 2009