UCI

La Habana, junio de 2010

Año del 52 Aniversario de la Revolución

Implementación del módulo “Técnica de ordenamiento de tarjetas” para la plataforma de arquitectura de la información “ABAD”

Autores:

José Gabriel Espinosa Ramírez (Facultad 9)

Yenier Jiménez Morales (Facultad 9)

Tutores:

Ing. Yanicet Aveleira Rodríguez.

Lic. Keyttia Pintón Almenares

Cotutores:

Lic. Rodrigo Ronda León

Ing. Eliober Cleger Despaigne

Dedicatoria

Dedico este trabajo a mi madre por el afecto que siempre me ha entregado y su constante atención hacia mi formación, a mi padre por ser mi ejemplo y guía. A mi hermano David, que siempre me sorprende por su constancia y entusiasmo. A mis tíos y abuelos por ser mi fuente de inspiración en los momentos más difíciles. A mi novia Milenis por su constante apoyo y comprensión. A Luz María, Adelina y Virginia, que con tanto cariño cuidaron de mí. A mis vecinos, compañeros de estudios y amigos.

José Gabriel Espinosa Ramírez.

Difícil es fijar en el poco espacio que brinda un papel lo mucho que varias personas ocupan en nuestras vidas. Este trabajo va dedicado a mi madre y mi padre por la educación brindada, por el amor entregado, por ser mis mejores maestros en la vida; a mi abuela por su cariño incansable; a mis hermanos por tener en ellos las mejores compañías; a toda mi familia por el apoyo ofrecido en todos estos años de estudio; a Yeney, mi esposa, por brindarme una sonrisa cuando no es el mejor de los momentos, por ser la mano amiga más cercana y por su infinito amor; Odalis y Jorge por asumirme como otro de sus hijo, a Julita y a su maravillosa familia; a mis vecinos, amigos; y a los que ya no están entre nosotros pero que me dieron su amistad y cariño.

Yenier Jiménez Morales.

Agradecimientos

El presente trabajo es el fruto del empeño de muchas personas. En primer lugar mis agradecimientos son para la Revolución Cubana y la Universidad de las Ciencias Informáticas por permitirme estudiar y formarme como profesional. Deseo expresar mi gratitud hacia el magnífico equipo de trabajo integrado por la Ing. Yanicet Aveleira, la Lic. Keyttia Pintón, la Ing. Dayaisis Bernis, el Lic. Rodrigo Ronda León, el Ing. Eliober Cleger y mi colega Yenier Jiménez. Asimismo agradecer de forma especial al profesor Lic. Asdrúbal Ramírez, a la Lic. Liz Mary Cruz, a la Lic. Marianela Curbelo, al Ing. David Silva, al Lic. Yusef Hassan, a Donna Spencer y a la editorial Rosenfeld Media por orientarnos a lo largo del proceso de desarrollo. Agradezco a todos los profesores y compañeros de estudio que a lo largo de estos años han contribuido con su ejemplo y conocimientos. Un especial agradecimiento a mis padres, a mi hermano David y a mi novia Milenis por haberme apoyado siempre.

José Gabriel Espinosa Ramírez.

Durante el desarrollo del presente trabajo de diploma convergieron los esfuerzos de varias personas. No se llega muy lejos en el desarrollo de sistemas informáticos si el trabajo en equipo no prevalece por encima de intereses y pretensiones personales; por eso quisiera en primer lugar agradecer al buen equipo de trabajo del cual formé parte integrado por la Ing. Yanicet Aveleira, la Lic. Keyttia Pintón, el Lic. Rodrigo Ronda León y mi compañero en la autoría del presente trabajo José Gabriel Espinosa. Agradezco a todos los profesores que intervinieron en mi proceso de formación docente y productiva, a Asdrúbal por guiarnos por los interesantes caminos de la investigación y tener la primera idea de hacer una herramienta como la desarrollada, a mis compañeros de estudios. De manera especial agradezco a cada persona que intervino en la creación y construcción de esta universidad que nos ha formado como ingenieros.

Yenier Jiménez Morales.

Resumen

La arquitectura de la información es la disciplina encargada de estudiar, analizar y organizar la disposición y estructuración de la información en cualquier medio, de tal forma que sea consultada por el usuario de manera intuitiva (Garrett, 2002). Entre las técnicas que se utilizan en esta disciplina está el ordenamiento de tarjetas[1], que tiene como objetivo la elaboración de recomendaciones para optimizar el diseño centrado en el usuario de un espacio informacional.

El módulo “Técnica de ordenamiento de tarjetas”, “KSort”, de la plataforma para la arquitectura de la información ABAD, es una aplicación de escritorio para la automatización de la técnica de ordenamiento de tarjetas. Esta solución informática permite la preparación, ejecución y evaluación de los ejercicios de ordenamiento de tarjetas. El sistema fue elaborado siguiendo los pasos propuestos por el Proceso Unificado de Desarrollo de Software y modelado a través del Lenguaje Unificado de Modelado haciendo uso de la herramienta de CASE Visual Paradigm. La aplicación fue implementada en el lenguaje de programación orientada a objetos y multiplataforma Java.

[1] Del inglés Card Sorting.

Contenido

Introducción

El ser humano constantemente establece asociaciones entre conceptos y de esta manera va construyendo su representación del universo que lo rodea. Estas asociaciones construyen patrones de agrupamiento bajo los cuales se van incorporando los nuevos conceptos, experiencias, conocimientos y todos aquellos elementos con los que interactúa a lo largo de su vida. Estos patrones pueden ser aprovechados para elaborar, estructurar y disponer la información del tal forma que le sea lo más útil posible.

Existe una disciplina llamada arquitectura de la información que es la encargada de estudiar, analizar y organizar la disposición y estructuración de la información en cualquier medio, de tal forma que sea consultada por el usuario de manera intuitiva (Garrett, 2002). Entre las técnicas que se utilizan en esta disciplina está el ordenamiento de tarjetas, que tiene como fin realizar recomendaciones para optimizar el diseño centrado en el usuario y de esta manera lograr una mejor interacción con el mismo.

El ordenamiento de tarjetas consiste en brindarles a los participantes[2] un conjunto de tarjetas[3] para que las agrupen según su criterio individual (Spencer, 2009). Posteriormente se analiza la composición de cada agrupamiento, obteniéndose como resultado un informe de recomendaciones para la organización de la información de acuerdo con los patrones de agrupamiento de los participantes. Esta técnica se caracteriza por la complejidad del análisis para una gran cantidad de participantes, por lo que su realización manual es engorrosa.

Situación problemática: En la Universidad de las Ciencias Informáticas (UCI) se ha iniciado el estudio de la arquitectura de la información con vistas a aplicarla directamente a los proyectos productivos. Los arquitectos de la información de este centro de estudios carecen de una solución informática para el ordenamiento de tarjetas, por lo que deben realizar sus procesos de forma manual y no estandarizada, lo que trae consigo inexactitud y demoras en el mismo.

[2] Segmento de la audiencia del producto final.
[3] Cada Tarjeta contiene un término y un concepto relacionados.

Problema científico: La realización del ordenamiento de tarjetas de forma manual atenta contra la agilidad, estandarización y precisión del mismo.

En la presente investigación el **objeto de estudio** lo constituye la Arquitectura de la Información.

Teniendo como **hipótesis**: “La automatización del ordenamiento de tarjetas incrementará la calidad, estandarización y agilidad del ordenamiento de tarjetas”.

El **campo de acción** abarcado es el ordenamiento de tarjetas.

Se ha identificado como **objetivo general** de la investigación desarrollar una solución informática que incremente la agilidad, calidad y estandarización del ordenamiento de tarjetas.

Como **objetivos específicos** se han identificado:

- Caracterizar el ordenamiento de tarjetas como proceso de la arquitectura de información.
- Elaborar una propuesta para la automatización del ordenamiento de tarjetas.
- Validar la solución mediante pruebas y los criterios de expertos.

Se ha identificado como posible **resultado a obtener** una solución informática para el ordenamiento de tarjetas acorde a las necesidades de los arquitectos de la información de la UCI, tributando con ello a la usabilidad de las soluciones informáticas que se desarrollen en los proyectos productivos de la misma.

El trabajo está estructurado en cuatro capítulos:

Capítulo 1. Fundamentación teórica:

Incluye un análisis del estado del arte a nivel internacional, nacional y de la Universidad, de las tendencias, técnicas, tecnologías, metodologías y software, en los que se apoya la investigación para la solución del problema. Son expuestos los principales elementos teóricos necesarios para la elaboración de la investigación. Finalmente, se proponen los elementos técnicos a utilizar en el desarrollo de la solución informática justificando su selección.

Capítulo 2. Características del sistema:

Describe el negocio a través de un modelo de negocio. Se detallan las reglas a considerar, los actores y trabajadores que intervienen, el modelo de casos de uso del negocio y el modelo de objetos. Se definen los requisitos, el modelo de casos de uso y los actores del sistema.

Capítulo 3. Construcción del sistema:

Describe la solución propuesta, basada en los requisitos identificados en el capítulo 2, centrándose en el diseño del sistema y refinándose la arquitectura del mismo.

Capítulo 4. Implementación y prueba:

Incluye la implementación y pruebas del sistema informático. Es descrita la solución informática en términos de componentes de software. Conjuntamente son definidas las pruebas a las que será sometido.

Antecedentes

La presente investigación tiene sus antecedentes en los trabajos de diploma: “Análisis de un sistema automatizado a través de la técnica de CardSorting u Ordenación de Tarjetas” (Sola Padilla, 2008), y “Análisis y Diseño de un sistema para la aplicación de técnicas de CardSorting en la obtención de Arquitecturas de Información” (Orovio Pino, 2009) y en las aplicaciones WebSort (Lime & Chile, 2009), OptimalSort (OptimalSort, 2009), SynCaps (Syntagm, 1995) y UXsort (UsabilityCentric, 2009).

Capítulo 1. Fundamentación teórica

La arquitectura de la información es la disciplina encargada del estudio, análisis, organización, disposición y estructuración de la información en espacios de información. En la actualidad la arquitectura de la información es muy utilizada en el diseño de las interfaces para sitios Web, aunque en sus inicios fue empleada en otros escenarios fuera de la Web.

Esta disciplina trata de organizar la información de manera que los usuarios puedan encontrar las respuestas correctas a sus interrogantes (Montes de Oca, 2004). Para esto, la arquitectura de la información se vale de varias técnicas entre las que se destacan los Métodos de Diseño Centrados en el Usuario y dentro de estos se encuentra el ordenamiento de tarjetas.

El ordenamiento de tarjetas es un Método de Diseño Centrado en el Usuario participativo, utilizado por los arquitectos de la información[4] para comprender cómo los usuarios reconocen y modelan la información (Paul, 2008). El ordenamiento de tarjetas puede proveer una vista interior de los modelos mentales del usuario, revelando la forma en que ellos tácitamente agrupan, ordenan y etiquetan tareas y contenidos dentro de sus mentes (Warfel, 2004).

El ordenamiento de tarjetas está compuesto por tres fases: preparación, ejecución y evaluación. En la primera fase se prepara el ejercicio de ordenamiento de tarjetas, mediante la elaboración de las instrucciones, las tarjetas[5] y las categorías[6]. Durante la fase de ejecución se le ofrecen las tarjetas y categorías a los participantes[7] para que las agrupen en clases[8] según su criterio individual (Spencer, 2009). En su última fase se evalúan las clases obtenidas mediante algoritmos de formación de agrupamientos, obteniéndose recomendaciones para la organización de la información de acuerdo con los patrones de los participantes.

[4] “Persona que crea el mapa o la estructura de información que permite a otros encontrar su camino personal al conocimiento” (Wurman, 1997).
[5] Cada Tarjeta contiene un término y un concepto relacionados.
[6] Espacio utilizado por los participantes para crear las clases durante la ejecución.
[7] Segmento de la audiencia del producto final.
[8] Agrupamiento de tarjetas creado por un participante sobre una categoría.

Existen dos variantes de ejercicios de ordenamiento de tarjetas: abierto[9] y cerrado[10]. Los ejercicios de ordenamiento de tarjetas abiertos son útiles para el posicionamiento de nuevas estructuras de información en un sitio Web o producto. Mientras que los ejercicios de ordenamiento de tarjetas cerrados son recomendados para agregar contenidos a una estructura existente u obtener retroalimentación adicional luego de realizado un ejercicio de ordenamiento de tarjetas abierto (Warfel, 2004).

El ordenamiento de tarjetas presenta aspectos ventajosos y desventajosos que se deben tener presente para determinar la técnica a utilizar y la forma en que esta se aplica:

Ventajas:

- **Simple**: El ordenamiento de tarjetas es simple para el organizador y los participantes.
- **Barata**: Usualmente el costo es un grupo de tarjetas de 3x5, un block de notas, una pluma o varias etiquetas impresas y el tiempo de las personas involucradas.
- **Rápida ejecución**: Se pueden realizar varios ordenamientos en un período de tiempo relativamente corto, los cuales proveerán un cúmulo significativo de datos.
- **Reconocimiento**: Esta técnica ha sido utilizada por más de 10 años por muchos diseñadores.
- **Involucra a los usuarios**: Porque debería ser más fácil de usar la estructura de información sugerida por el ordenamiento de tarjetas, que está basada en los modelos mentales de usuarios reales, no en los instintos o sólidas opiniones de un diseñador, arquitecto de la información, o de un cliente clave.
- **Provee buenos fundamentos**: No es la mejor solución pero suministra buenos fundamentos para la estructura de un sitio Web o producto (Warfel, 2004).

Desventajas:

- **No considera los roles del usuario:** El ordenamiento de tarjetas es una técnica de naturaleza centrada en el contenido. Si es utilizada sin tener presente los roles de los usuarios, puede

[9] En el cual los participantes pueden crear nuevas categorías.
[10] En el cual los participantes no pueden crear nuevas categorías.

encaminarse hacia una estructura de información que no sea la más indicada ante los roles que deben desempeñar los usuarios. Es necesario un análisis de roles o de necesidades de información para asegurar que el contenido a ordenar coincida con las necesidades de los usuarios y que la estructura de información resultante permita a los usuarios cumplir sus roles.

- **Los resultados pueden variar:** El ordenamiento de tarjetas puede proveer aleatoriamente resultados similares entre los participantes, o estos pueden ser muy diferentes entre sí.
- **El análisis puede consumir bastante tiempo:** El ordenamiento es rápido, pero el análisis de los datos puede ser dificultoso y consumir bastante tiempo, particularmente si los resultados de los participantes son muy diferentes.
- **Puede ser que capture solo características superficiales:** Los participantes pueden no considerar sobre lo que trata el contenido o cómo lo utilizarán y puede suceder que ordenen las tarjetas por sus características superficiales como si fueran tipos de documentos (Warfel, 2004).

Para la evaluación de los ejercicios de ordenamiento de tarjetas se transforman las clases y tarjetas en vectores que serán utilizados por los algoritmos de formación de agrupamientos planos y jerárquicos. La meta de estos algoritmos es crear agrupamientos coherentes internamente pero claramente diferenciados de otros agrupamientos. En otras palabras, los elementos de un mismo agrupamiento deben ser tan similares como sea posible y los elementos de diferentes agrupamientos deben ser tan disímiles como sea posible (Manning, 2007).

Dentro de los algoritmos de formación de agrupamientos planos (ω) se encuentra el K – Means, cuyo objetivo es minimizar el cuadrado de la distancia euclidiana de los elementos del agrupamiento al centro del mismo. Este centro es definido como la media o el centroide ($\vec{\mu}$) y está determinado por la siguiente ecuación (Manning, 2007):

$$\vec{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\vec{x} \in \omega} \vec{x}$$

El primer paso en el K – Means es seleccionar al azar como centros de los agrupamientos K elementos, llamados semillas. Luego iterativamente se agregan los elementos al agrupamiento más cercano al mismo. Se calculan nuevamente todos los centros de los agrupamientos o centroides. Se comprueba la

condición de parada, en caso de que esta se cumpla se culmina el algoritmo, en caso contrario se agregan nuevamente los elementos a los clústeres[11] más cercanos, se calculan los centros de agrupamiento y se comprueba nuevamente la condición de parada, la cual está determinada por:

- Un número de iteraciones completadas: Esta condición limita el tiempo de ejecución del algoritmo, en algunos casos puede provocar que la calidad de los agrupamientos sea pobre.
- La composición de los agrupamientos no varía entre iteraciones: A excepción de los casos en los que se obtienen mínimos locales, suele indicar que se ha obtenido un buen agrupamiento; pero en ocasiones el tiempo de ejecución es extremadamente largo.
- Los centroides no varían entre iteraciones: Es equivalente a que la composición de los agrupamientos se mantenga invariable.
- La función suma residual de cuadrados (SRC) queda por debajo de un valor determinado: Esto garantiza una calidad mínima deseada en los agrupamientos antes de terminar. En la práctica se recomienda utilizar este criterio junto con el de la cantidad de iteraciones para garantizar un tiempo de ejecución y resultados aceptables.
- El decrecimiento del resultado de la función SRC queda por debajo de un valor deseado: Para pequeños decrecimientos se indica que se está cerca de la convergencia con un agrupamiento óptimo. Nuevamente se recomienda utilizar este criterio, junto con el del número de iteraciones, para evitar tiempos de ejecución prolongados (Manning2, 2007).

$$SRC = \sum_{x=1}^{K} SRC_x$$

$$SRC_x(\vec{v}) = \sum_{\vec{j} \epsilon \omega_x} |\vec{v} - \vec{j}|^2 = \sum_{\vec{j} \epsilon \omega_x} \sum_{m=1}^{M} (v_m - x_m)^2$$

$$\frac{\partial SRC_k(\vec{v})}{\partial v_m} = \sum_{x \in \omega_k} 2(v_m - x_m)$$

[11] Agrupamientos.

Donde x_m y v_m son los elementos número m de sus respectivos vectores y $\vec{v}$ es vector del centroide del agrupamiento. La función SRC se minimiza al remplazarse los centroides, por lo que la suma total de la función SRC para todos los agrupamientos también decrece (Manning, 2007).

Dentro de los algoritmos de formación de agrupamientos jerárquicos se tienen dos tipos principales: tope – base y los base – tope. Los algoritmos base – tope toman cada elemento como un agrupamiento independiente y en iteraciones sucesivas mezcla el par de agrupamientos más cercanos hasta que todos queden incluidos en un solo agrupamiento que contiene a todos los elementos. Los algoritmos tope – base requieren de un método recursivo que divida los agrupamientos, partiendo de uno que contenga a todos los elementos, hasta que se llegue a un elemento indivisible (Manning, 2007).

Los algoritmos de formación de agrupamientos jerárquicos más utilizados son: Single – Link, Complete – Link, Group – Average y Centroid – Clustering. Se diferencian en el criterio de similitud que se toma para formar los agrupamientos (Manning, 2007).

Tabla 1. Criterios de similitud para algoritmos jerárquicos.

Algoritmo	***Criterio***
Single – Link	El par de elementos más cercanos de estos.
Complete – Link	El par de elementos más diferentes de estos.
Group – Average	La similitud promedio entre todos los pares de elementos, incluidos aquellos del mismo agrupamiento.
Centroid – Clustering	Semejanza de sus centroides.

La UCI ha comenzado a aplicar la arquitectura de la información a sus proyectos productivos. Como parte de este proceso, ha decidido desarrollar una plataforma de herramientas para la Arquitectura de la Información, la cual se denominará ABAD.

Abad es el título con que se designa al sacerdote principal de una Abadía, monasterio con tal rango. Durante la edad media en los monasterios se concentró el conocimiento de la humanidad; por lo que entre las principales tareas del Abad estaba velar por la información y sabiduría bajo la custodia de su abadía. Inspirados en la historia se determinó que la plataforma de herramientas para la arquitectura de la información que se desarrollará en el seno de la UCI se denominará ABAD.

Esta plataforma necesita una herramienta para desarrollar el ordenamiento de tarjetas. La misma debe ser una aplicación de escritorio, compatible con las otras herramientas de dicha plataforma. Debe permitir diseñar ejercicios de ordenamiento de tarjetas, ejecutarlos y analizar los resultados mediante algoritmos de formación de agrupamientos.

Existen varias herramientas para realizar el ordenamiento de tarjetas, entre las cuales sobresalen el WebSort (Lime & Chile, 2009), el OptimalSort (OptimalSort, 2009), el SynCaps V2 de Syntagm (Syntagm, 1995) y el UXSort (UsabilityCentric, 2009).

El WebSort y el OptimalSort son aplicaciones Web que permiten realizar el ordenamiento de tarjetas desde Internet así como procesar datos de los agrupamientos realizados por los participantes en ejercicios anteriores. El OptimalSort fue desarrollado por la compañía Neozelandesa OptimalUsability en el año 2004. OptimalUsability es una consultora que vio frustrados varios de sus proyectos de arquitectura de la información producto de las limitaciones de las herramientas para la realización del ordenamiento de tarjetas. Por lo que en el 2004 construyeron su propia aplicación (OptimalSort, 2009).

El WebSort es una aplicación Web para la realización del ordenamiento de tarjetas desarrollada por la compañía Lime & Chile Productions. El WebSort se distingue por poseer una interfaz de usuario elegante, brindar varias vistas de los resultados, y la compatibilidad de estos con Microsoft Excel, así como la capacidad de procesar datos procedentes de otras herramientas para realizar el ordenamiento de tarjetas tales como el SynCaps V2 de Syntagm.

Tanto el WebSort como el OptimalSort son aplicaciones privativas que operan desde la Web por lo que no se puede garantizar la compatibilidad de estas con el resto de las herramientas de la plataforma ABAD.

El SynCaps (Computer Aided Paper Sorting) V2, es una solución creada por la compañía británica SyntagmLimited para el análisis de los resultados de los ejercicios de ordenamiento de tarjetas. Syntagm fue fundada por William Hudson en 1985 con la visión de proveer nuevos servicios de desarrollo de software para los nuevos fundamentos de las interfaces gráficas de usuarios, servir de consultores para temas relacionados con la usabilidad, así como impartir cursos sobre temas relacionados (Syntagm, 1995). El SynCaps es una aplicación de escritorio, privativa, y orientada para la fase de evaluación;

excluye las dos primeras fases del ordenamiento de tarjetas y tiene restringido el acceso a su código fuente, por lo que no es posible garantizar la compatibilidad de la misma con la plataforma ABAD.

El UXsort es una solución informática para la realización del ordenamiento de tarjetas que permite a los profesionales de la usabilidad y usuarios avanzados planear ejercicios de ordenamiento de tarjetas, gestionar los participantes y tarjetas, almacenar los datos de los ordenamientos realizados por los participantes y crear un reporte final (UsabilityCentric, 2009). El UXsort fue creado haciendo uso del lenguaje de programación orientado a objetos C# y del marco de trabajo Microsoft .Net, por lo que no es multiplataforma, ni compatible con la plataforma ABAD.

Tendencias y tecnologías a considerar

Metodologías de desarrollo de software

Las metodologías de desarrollo de software definen un conjunto de procesos y actividades que guían a los desarrolladores de software durante el ciclo de vida de un proyecto, especifican los artefactos a construir y organizan el personal involucrado en roles dentro del equipo de trabajo. Existen dos grandes grupos de metodologías: las ágiles y las robustas o tradicionales.

Metodologías ágiles de desarrollo de software

Las metodologías ágiles o ligeras se distinguen por la simplicidad de sus reglas, prácticas, su orientación a equipos de desarrollo de pequeño tamaño, su flexibilidad ante los cambios y su ideología de la colaboración constante. Este tipo de metodologías define pocos artefactos a generar y roles que asumir en el proceso de desarrollo de software. Identifica al cliente como un miembro activo más dentro del equipo de desarrollo. Están concebidas para proyectos donde los cambios en el desarrollo del sistema se produzcan con mucha frecuencia.

eXtremeProgramming, (XP) ó Programación Extrema.

Promueve, para el éxito del desarrollo de un software, las relaciones interpersonales, así como el continuo aprendizaje de los miembros del equipo de trabajo y el fomento de la retroalimentación entre clientes y

desarrolladores. Según esta metodología el cliente debe estar presente en todo momento junto al equipo de desarrollo en el proyecto pues tiene la responsabilidad de orientar el trabajo hacia lo que aporte mayor valor al negocio. El ciclo de vida en XP queda definido en seis fases: Exploración, Planificación de la Entrega, Iteraciones, Producción, Mantenimiento y Muerte del Proyecto (Kent Beck, 1999).

Durante el desarrollo de la plataforma ABAD se esperan cambios en el equipo de desarrollo, por lo que se requiere que el proceso esté sólidamente documentado. No se esperan cambios frecuentes de los requisitos y no se puede garantizar la presencia constante del cliente, por lo que esta metodología no será adoptada.

Metodologías robustas de desarrollo de software.

Las metodologías robustas de desarrollo de software o metodologías tradicionales surgen fruto de la acumulación de experiencias de determinadas empresas en la construcción de sistemas informáticos. A diferencia del nuevo enfoque ágil, ofrecen resistencia ante los cambios solicitados por un cliente. El equipo de trabajo conformado es mayor que el propuesto por las alternativas ágiles. Los artefactos generados son sustantivamente más numerosos que en las metodologías ágiles. Las metodologías robustas sugieren mantener un proceso lo más organizado y gestionado posible, evitando que la pérdida de personal dentro del grupo de desarrollo provoque que el proceso se vea afectado.

Proceso Unificado de Desarrollo.

“El Proceso Unificado de Desarrollo (en lo adelante RUP) es un marco genérico que puede especializarse para una gran variedad de sistemas, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos. (…) RUP utiliza el Lenguaje Unificado de Modelado (Unified Modeling Lenguague, en lo adelante UML) como lenguaje de modelado” (Jacobson, et al., 2000).

RUP se caracteriza por ser dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. El proceso de desarrollo de software se organiza a través de nueve flujos de trabajo organizados en cuatro fases de desarrollo. Los flujos de trabajo que define son: Modelado del Negocio, Levantamiento de

Requisitos, Análisis y Diseño, Implementación, Prueba, Instalación, Gestión de Proyecto, Gestión de Configuración y Cambios y Ambiente. Define cuatro fases por las que todo proyecto informático debe ubicarse gradualmente: Inicio, Elaboración, Construcción y Transición (Jacobson, et al., 2000).

Entre el equipo de desarrollo y los clientes no existe un alto grado de incertidumbre en cuanto a las funcionalidades que el sistema debe cumplir. Los requisitos identificados no cambiarán, debido a que los procedimientos y algoritmos de evaluación de datos del ordenamiento de tarjetas son estables. Al establecer que la construcción esté basada en componentes de software, facilitará el seguimiento de los componentes construidos y una mayor organización del equipo de desarrollo. Permitirá mantener el proceso de desarrollo sólidamente documentado, disminuyendo el impacto de los cambios en el equipo de desarrollo. Por lo que se propone utilizar esta metodología.

Aplicaciones de escritorio y aplicaciones Web

Los sistemas informáticos pueden ser clasificados como aplicaciones de escritorio, aplicaciones Web, aplicaciones por consola, entre otras muchas categorías. Estas clasificaciones responden a la forma en la cual un software se distribuye, a la manera en que se produce el flujo de datos en el mismo, el modo de instalación, y de procesamiento de los datos. Las dos categorías más afines a los propósitos del presente trabajo son las aplicaciones de escritorio y las aplicaciones Web.

“Una aplicación Web consiste en una página Web con contenido dinámico, es decir, que cambia su contenido según la interacción de cada usuario que la visita” (Ailonwebs, 2009). Sus usuarios no necesitan instalar el software en sus computadoras sino tener instalado un navegador Web para visualizarlo y una red que permita la conexión a donde la aplicación de se encuentre alojada. Una aplicación de escritorio a diferencia de las aplicaciones Web consiste en un software almacenado y ejecutado directamente en las computadoras de sus usuarios.

Los usuarios finales de una solución informática a los cuales se les aplicará el ordenamiento de tarjetas no siempre pudieran contar en su institución con una red de computadoras, por lo que un sistema que automatice la técnica a través de una aplicación Web no sería apropiado. Además, se debe tener en cuenta que la plataforma ABAD será una aplicación de escritorio. Por lo que se propone desarrollar el

sistema como una aplicación de escritorio debido a las características del entorno en el cual se aplicará el sistema y a la plataforma con la que el mismo se integrará.

Lenguajes de Programación

Python

Python es un lenguaje interpretado que forma parte de un proyecto de software libre. Al no ser compilado como muchos otros lenguajes, su código no produce un ejecutable, sino que existe un intérprete que ejecutará el código sobre una plataforma específica (Python, 2009). Ha incorporado gran cantidad de librerías. Dispone de varias funciones para el tratamiento de las cadenas de texto, números, archivos. Se pueden importar librerías en los programas para tratar temas específicos como la programación de ventanas. Por ser un lenguaje de alto nivel y al tratarse de un lenguaje interpretado, el rendimiento es más bajo; por lo que no se adoptará para la elaboración de esta solución informática.

C++

C++ es un lenguaje imperativo[12] orientado a objetos, basado en su predecesor, C. Al igual que C, su ejecución sigue muy ligada al hardware subyacente manteniendo una considerable potencia para la programación a bajo nivel, aunque también permite un estilo de programación con un alto nivel de abstracción.

"La velocidad de ejecución de C++ es superior a muchos lenguajes. No es orientado a objeto puro si se compara con otros lenguajes, por ejemplo Java; aunque introduce nuevas palabras claves y operadores para el manejo de clases. Algunas de sus extensiones tienen aplicación fuera del contexto de programación orientada a objetos pues muchos aspectos pueden ser usados independientemente de las clases" (Soulie, 2009). Al estar ligado al hardware subyacente, las soluciones desarrolladas en C++ deben ser recompiladas para cada plataforma y presentan limitaciones para la escalabilidad hacia la Web. Por lo que no se adoptará para la elaboración de esta solución informática.

[12] Especifican cómo conseguir los objetivos que se persigue.

Java

Java constituye un lenguaje eficaz y muy versátil. Permite a los desarrolladores crear un sistema informático en una plataforma y ejecutarlo en otra distinta de la inicial. Se caracteriza por su potencia, y a la vez elimina las características menos usadas y más complejas de otros lenguajes como C y C++.

Fue creado por Sun Microsystems. La mayor parte del código Java se encuentra bajo licencia GPL – GNU excepto las bibliotecas de clases de Sun. La tecnología Java está compuesta por dos partes fundamentales, el lenguaje de programación y la plataforma de desarrollo. El código Java una vez compilado genera un fichero que almacena el Bytecode[13]. Para su ejecución es necesario contar con una Máquina Virtual de Java (JVM), que consiste en un intérprete el cual lee el Bytecode y lo traduce en instrucciones ejecutables. "Los programas hechos con Java se caracterizan por ser orientados a objetos, distribuidos, interpretados, robustos, seguros, de arquitectura neutral, portátiles, de gran rendimiento, multitareas y dinámicos" (Zukowski, 2003).

Se propone utilizar Java como lenguaje en el desarrollo de esta solución informática para que la misma sea multiplataforma. Java permite el diseño y construcción de aplicaciones de múltiples propósitos, cuenta con librerías bien documentadas para la gestión de interfaces de usuario y facilita la escalabilidad hacia la Web.

Marcos de trabajo para la gestión de interfaces de usuario en Java

XForms

"XForms es un formato XML para construir interfaces de usuario Web, principalmente formularios Web. Ha sido diseñado para ser la nueva generación de formularios HTML/XHTML, pero es lo suficientemente genérico como para que pueda ser usado, de una manera independiente, para describir cualquier interfaz de usuario e incluso para realizar tareas simples y comunes de manipulación de datos" (OPTIC , 2009). Dado que la solución propuesta es una aplicación de escritorio no es recomendable utilizar este marco de trabajo para la gestión de interfaces de usuarios orientado a soluciones Web.

[13] Pseudocódigo prácticamente al nivel de código máquina.

Standard Widget Toolkit (SWT)

Inicialmente desarrollado por IBM y actualmente mantenido por la Fundación Eclipse, SWT es un marco de trabajo de código abierto diseñado para proveer interfaces de usuarios portables y eficientes (Eclipse, 2009). Aunque está implementado en Java se basa en componentes nativos de cada sistema operativo, lo que conlleva a que cada implementación sea única para cada plataforma. SWT varía su rendimiento según sea la plataforma utilizada y demanda de una gestión explícita de la destrucción de objetos, en oposición a los estándares de Java (Marinilli, 2006), por lo que no será utilizado para el desarrollo de esta solución.

AWT/Swing

Java proporciona una variante de creación de interfaces gráficas de usuario mediante paquetes estándares. Dos de ellos son AWT y Swing. AWT brinda componentes de entrada y salida de datos. La biblioteca AWT está ideada para funcionar como una API que utiliza los componentes nativos de un sistema operativo. Solo se puede disponer de las funcionalidades comunes en todos los sistemas operativos provocando la dependencia de la plataforma, sin embargo, la vinculación AWT/Swing elimina dicha dependencia. Swing amplía AWT adicionando un conjunto de componentes escritos en Java. AWT se encarga de la gestión de eventos y Swing brinda nuevos componentes.

Teniendo en cuenta que se debe desarrollar una aplicación de escritorio el marco de trabajo AWT/Swing es el más apropiado pues está orientado a la construcción de aplicaciones de escritorio que requieran de una considerable gestión de interfaces de usuario. Es nativo de Java por lo que para su uso no es necesario adquirir licencia alguna.

Ambiente de Desarrollo Integrado (IDE)[14]

Para el desarrollo del sistema se identificaron como posibles IDEs a utilizar el Eclipse 3.1 y NetBeans 6.8 M2. Eclipse es un IDE de las plataformas enriquecidas (RCP). Esta característica viene dada por su

[14] Integrated Development Environment

capacidad de extensibilidad a través de la integración de módulos con fines específicos (Under, 2009). El componente definido en Eclipse para la construcción de interfaces gráficas es el Standard Widget Toolkit (SWT), que constituye una restricción para el desarrollo en este IDE pues se definió como marco de trabajo para la gestión de las interfaces gráficas de usuario el AWT/Swing.

Al igual que Eclipse NetBeans 6.8 M2 es un paquete completo y enriquecido. El mismo incluye un gran número de módulos para el desarrollo del sistema informático, sin necesidad de complementos. En caso de escalar el sistema hacia la Web la utilización de NetBeans 6.8 M2 ofrece mayores ventajas pues brinda un mayor soporte a la tecnología JavaServer Faces (JSF), encargada de simplificar el desarrollo de interfaces gráficas de usuario a través de JavaServer Pages como su tecnología de despliegue.

El desarrollo del sistema propuesto se realizará sobre el IDE NetBeans 6.8 M2 basado en las funcionalidades que ofrece para escribir, compilar, depurar y ejecutar programas escritos en Java. Es distribuido de manera libre, sin restricciones de uso. El desarrollador obtiene todas las herramientas necesarias para la creación de aplicaciones profesionales de escritorio, empresarial y para terminales móviles. Se ejecuta sobre varias plataformas (Windows, Mac OS X, GNU/Linux y Solaris). Facilita la gestión de las interfaces de usuario, la administración de las configuraciones del usuario y gestión de almacenamiento.

Lenguaje Unificado de Modelado

Partiendo de la propuesta de adoptar RUP como metodología de desarrollo se modelará el sistema a través de UML. El cual es un lenguaje de modelado que permite describir, documentar, visualizar y desarrollar un sistema informático orientado a objetos. Fue creado para elaborar una notación única de modelado que unificara todas las técnicas de modelado existentes en el momento de su nacimiento.

“UML ayuda al usuario a entender la realidad de la tecnología y la posibilidad de que reflexione antes de invertir y gastar grandes cantidades en proyectos que no estén seguros en su desarrollo, reduciendo el coste y el tiempo empleado en la construcción de las piezas que constituirán el modelo” (Jacobson, et al., 2000).

Modelado de software

Las herramientas CASE[15] constituyen sistemas para el modelado de software. En el mercado de las herramientas CASE se identifican las dos más usadas en el modelado de sistemas, Rational Rose Enterprise Edition y Visual Paradigm (VP). Tanto Rational Rose como VP facilitan la organización de los modelos de acuerdo con el orden lógico que propone RUP. VP permite visualizar, diseñar y documentar diagramas de UML 2.1 en un entorno de diseño intuitivo (Visual Paradigm, 2010). Mientras que el Rational Rose Enterprise Edition solo permite el diseño de diagramas de UML 1.x (IBM, 2010).

VP ayuda al equipo de desarrollo a controlar el progreso del proyecto y brinda un medio de comunicación. Facilita el modelado de un sistema donde cada involucrado puede crear sus propias vistas arquitectónicas (Vista Lógica, Vista de Procesos, Vista de Desarrollo, Vista Física y Vista de Casos de Uso). Ante cambios que surjan en un negocio, VP permite adaptar rápidamente los modelos realizados a dichos cambios, lo cual evita escribir código sin analizar los cambios en el modelo. Con VP es posible descomponer un modelado de un sistema en unidades controladas, generar código en un lenguaje específico a partir de los diseños realizados. Por todo lo anterior se optó por modelar la solución informática a construir con VP – UML 6.4 Enterprise Edition.

Conclusiones del capítulo

La ejecución manual de los algoritmos de evaluación utilizados por la técnica de ordenamiento de tarjetas es compleja. Las herramientas para el ordenamiento de tarjetas disponibles en el mercado no cumplen los requisitos necesarios para ser integradas con la plataforma ABAD, por lo que es necesario desarrollar una solución que se ajuste a estos requerimientos. El proceso de desarrollo del sistema informático será guiado a través de la metodología RUP y estará basado en tecnologías multiplataforma.

[15] ***C****omputer* ***A****ided* ***S****oftware* ***E****ngineering*, Ingeniería de Software Asistida por Ordenador

Capítulo 2. Características del sistema

Reglas del negocio

Las reglas del negocio describen las políticas o condiciones que deben ser satisfechas en el negocio. En el proceso de ordenamiento de tarjetas están presentes las siguientes reglas:

- Existen dos tipos de ejercicios de ordenamiento de tarjetas:
  - Ejercicio de ordenamiento de tarjetas abierto: los participantes pueden crear nuevas categorías.
  - Ejercicio de ordenamiento de tarjetas cerrado: los participantes no pueden crear nuevas categorías.
- Una tarjeta solo puede estar agrupada en una sola clase en cada solución de un ejercicio.
- No puede existir más de una clase con el mismo nombre en cada solución de un ejercicio.
- No puede existir más de una tarjeta con el mismo nombre en cada ejercicio.
- No puede existir más de una categoría con el mismo nombre en cada ejercicio.
- Los participantes no pueden eliminar las categorías creadas por el diseñador de ejercicios de ordenamiento de tarjetas.

Actores y trabajadores del negocio

Actor	Descripción
Arquitecto de la información	Persona encargada de estructurar la información en la interfaz de usuario de un producto determinado, para lo cual precisa de recomendaciones sobre la organización de la información.
Trabajador	**Descripción**
Diseñador de ejercicios de ordenamiento de tarjetas	Persona encargada de diseñar los ejercicios de ordenamiento de tarjetas.
Ejecutor de ejercicios de ordenamiento de tarjetas	Persona encargada de orientar y recopilar los resultados de la ejecución de un ejercicio de ordenamiento de tarjetas.
Participante	Miembro de la audiencia del producto final. Son los encargados de crear los agrupamientos de las tarjetas.
Evaluador de ejercicios de ordenamiento de tarjetas	Persona encargada de analizar y elaborar el informe de recomendaciones para la organización de la información.

Descripción de los Casos de Uso del Modelo del Negocio Actual

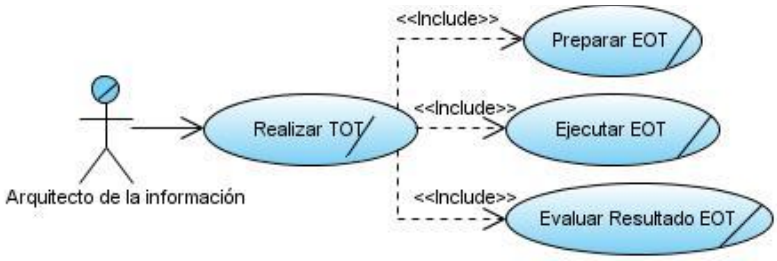


Fig. 1. Diagrama de casos de uso del negocio

Caso de Uso del Negocio Realizar Técnica de Ordenamiento de Tarjetas

Caso de Uso del Negocio	Realizar técnica de ordenamiento de tarjetas
Actores	Arquitecto de la información.
Resumen	El arquitecto de la información realiza una solicitud de recomendaciones para la organización de la información. El diseñador de ejercicios de ordenamiento de tarjetas prepara el ejercicio de ordenamiento de tarjetas (EOT). Los participantes ejecutan el EOT generando las clases. Finalmente, el evaluador de EOT analiza la composición de las clases y prepara un informe de recomendaciones para la organización de la información.
Casos de Uso asociados	Este caso de uso incluye los casos de usos: Preparar EOT, Ejecutar EOT y Evaluar Resultados EOT.
Curso normal de eventos	
Acción del actor	**Respuesta del proceso de negocio**
1. Realiza una solicitud de recomendaciones para la organización de la información.	1.1 Ir al curso normal de eventos de la descripción del CUN[16] Preparar EOT.
	1.2 Ir al curso normal de eventos de la descripción del CUN Ejecutar EOT.
	1.3 Ir al curso normal de eventos de la descripción del CUN Evaluar Resultados EOT.
Post condición	
Se obtiene un informe con las recomendaciones para la organización de la información.	

[16] Caso de Uso del Negocio.

Caso de Uso del Negocio Preparar Ejercicio de Ordenamiento de Tarjetas

Caso de Uso del Negocio	Preparar EOT
Actores	Arquitecto de la información.
Resumen	El arquitecto de la información realiza una solicitud de recomendaciones para la organización de la información. El diseñador de EOT prepara el EOT.
Casos de Uso asociados	Está incluido en el caso de uso del negocio Realizar TOT.
Curso normal de eventos	
Acción del actor	**Respuesta del proceso de negocio**
1. Realiza una solicitud de recomendaciones para la organización de la información.	1.1 Valida la solicitud. En caso de que la solicitud no esté correcta ir al flujo alternativo 1.
	1.2 Se crean las tarjetas.
	1.3 En caso de ser un EOT con categorías predefinidas ir al flujo alternativo 2.
	1.4 Se preparan las instrucciones y se finaliza el ejercicio.
Flujo alternativo 1	
	1.1 Notificar los errores.
	1.2 Finaliza el caso de uso.
Flujo alternativo 2	
	1.1 Se crean las categorías.
	1.2 Se regresa a la transición 1.4 del curso normal de eventos.
Post condición	
Se obtiene un EOT preparado	

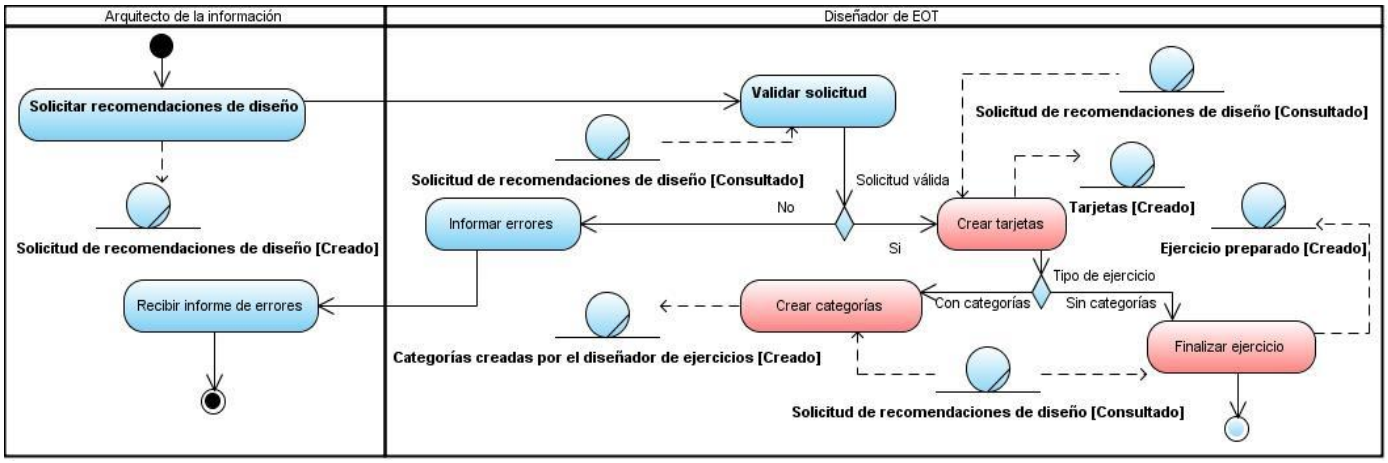


Fig. 2. Diagrama de actividades del CU Preparar Ejercicio de Ordenamiento de Tarjetas.

Caso de Uso del Negocio Ejecutar de Ejercicio de Ordenamiento de Tarjetas

Caso de Uso del Negocio	Ejecutar EOT
Precondición	
Se requiere de un EOT preparado y de los participantes.	
Resumen	Se inicia cuando los participantes son instruidos sobre el EOT y este le es entregado. Los participantes ejecutan el ejercicio. Se elabora un informe de los resultados.
Casos de Uso asociados	Está incluido en el CUN Realizar TOT.
Curso normal de eventos	
Acción del actor	**Respuesta del proceso de negocio**
	1.1 El ejecutor de ejercicios de ordenamiento de tarjetas instruye a los participantes.
	1.2 El participante ejecuta el ejercicio. En caso de ser un EOT abierto ir al flujo alternativo 1.
	1.3 Los participantes crean las clases agrupando las tarjetas sobre las categorías.
	1.4 Se elabora un informe con el compendio de las clases generadas por los participantes.
Flujo alternativo 1	
	1.1 El participante crea categorías.
	1.2 Regresa al curso normal de eventos en la transacción 1.3.
Post condición	
Se obtiene un informe de resultados con las clases generadas por los participantes.	

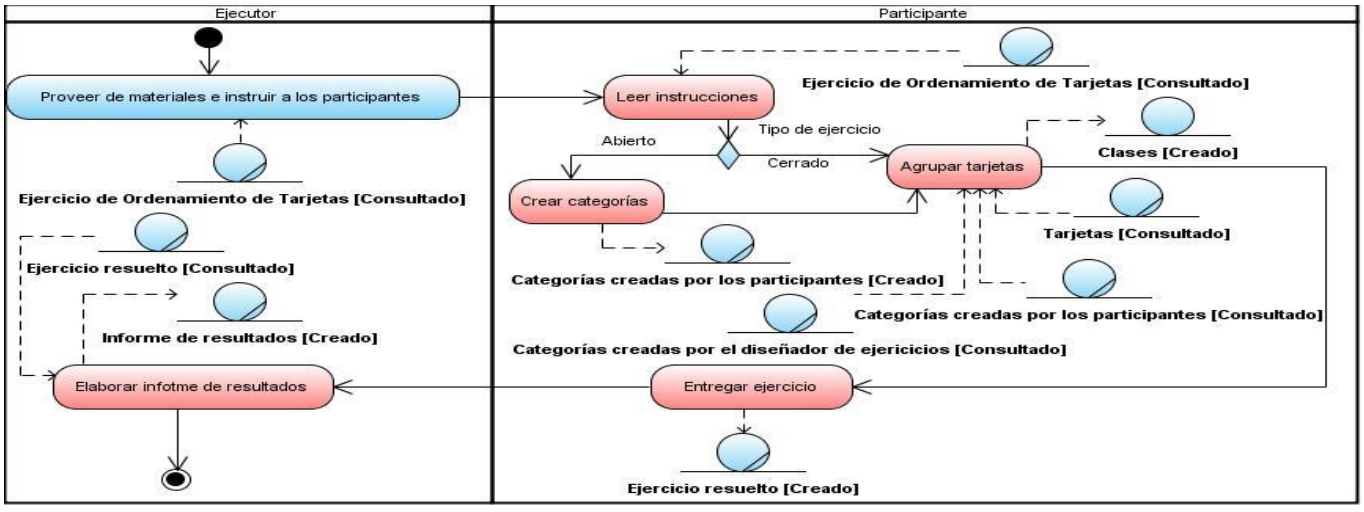


Fig. 3. Diagrama de actividades del CU Ejecutar Ejercicio de ordenamiento de tarjetas.

Caso de Uso del Negocio Evaluar Resultados del Ejercicio de Ordenamiento de Tarjetas

Caso de Uso del Negocio	Evaluar Resultados del Ejercicio de ordenamiento de tarjetas
Precondición	
Se requiere contar previamente con el informe de resultados de la ejecución de un EOT.	
Resumen	Se analizan los agrupamientos realizados por los participantes para obtener las recomendaciones de organización de la información.
Casos de Uso asociados	Está incluido en el caso de uso del negocio Realizar Técnica de Ordenamiento de Tarjetas.
Acción del actor	**Respuesta del proceso de negocio**
	1.1 Se tabulan los datos de las soluciones del EOT.
	1.2 Si se desea aplicar K-Means ir al flujo alternativo 1.
	1.3 Ir al flujo alternativo 2.
	1.4 Se elabora un informe final de recomendaciones para la organización de la información.
Flujo alternativo 1	
	1.1 Se aplica el algoritmo de formación de clústeres K – Means.
	1.2 Si se desea aplicar el algoritmo de aglomerados jerárquicos ir al flujo alternativo 2.
	1.2 Se regresa al curso normal de eventos en la transacción 1.4.
Flujo alternativo 2	
	1.1 Se aplica el algoritmo de aglomerados jerárquicos.
	1.2 Se regresa al curso normal de eventos en la transacción 1.4.
Post condición	
Se obtiene un informe con las recomendaciones para la organización de la información.	

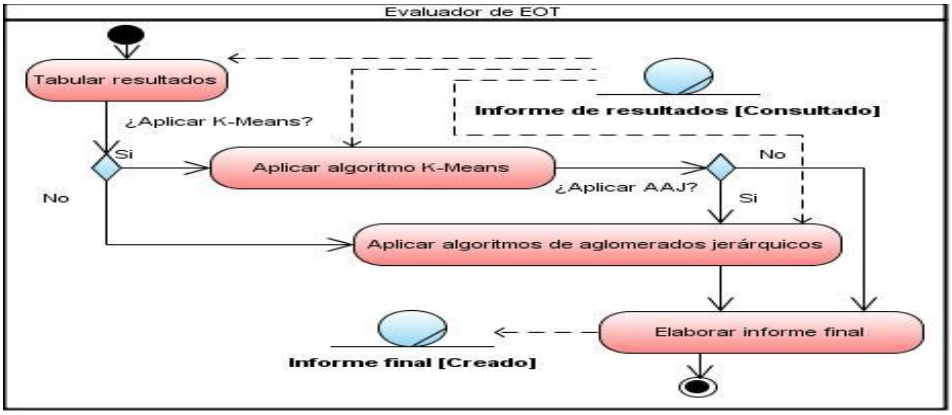


Fig. 4. Diagrama de actividades del CUN Evaluar Resultados de EOT.

Diagrama de clases del Modelo de Objetos

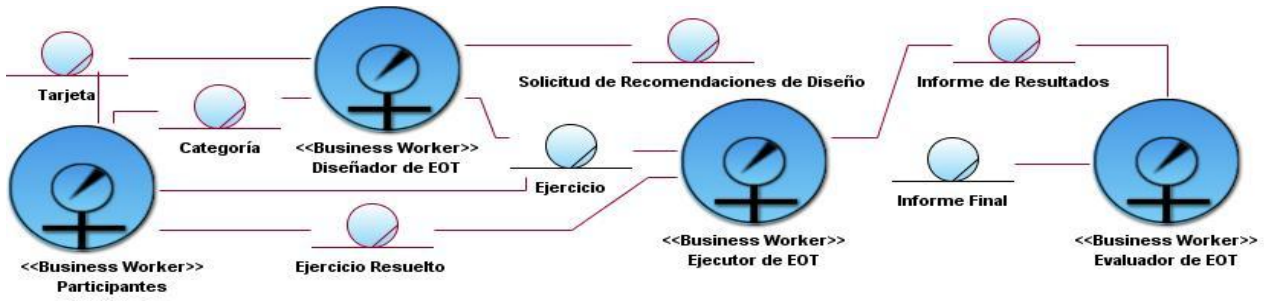


Fig. 5. Modelo de Objetos.

Requerimientos Funcionales

RF No.	Descripción
RF 1	Crear un ejercicio de ordenamiento de tarjetas.
RF 1.1	Crear un ejercicio de ordenamiento de tarjetas abierto.
RF 1.2	Crear un ejercicio de ordenamiento de tarjetas cerrado.
RF 2	Mostrar un ejercicio de ordenamiento de tarjetas y sus componentes.
RF 3	Almacenar el diseño de un ejercicio de ordenamiento de tarjetas.
RF 4	Editar un ejercicio de ordenamiento de tarjetas.
RF 4.1	Crear una tarjeta.
RF 4.2	Actualizar el contenido de una tarjeta.
RF 4.3	Eliminar una tarjeta.
RF 4.4	Crear una categoría.
RF 4.5	Actualizar el contenido de una categoría
RF 4.6	Eliminar una categoría
RF 5	Ejecutar un ejercicio de ordenamiento de tarjetas
RF 5.1	Ejecutar un ejercicio de ordenamiento de tarjetas abierto.
RF 5.1.1	Crear una clase por parte de los participantes.
RF 5.1.2	Editar el contenido de las clases creadas por parte de los participantes.
RF 5.1.3	Eliminar una clase creada por parte de los participantes.
RF 5.2	Ejecutar un ejercicio de ordenamiento de tarjetas cerrado.
RF 5.2.1	Agrupar tarjeta.
RF 5.2.2	Desagrupar tarjeta.
RF 6	Almacenar un ejercicio de ordenamiento de tarjetas ejecutado.
RF 7	Modificar ejercicio de ordenamiento de tarjetas ejecutado.
RF 8	Mezclar los Resultados de ejercicios de ordenamiento de tarjetas ejecutados.
RF 9	Elaborar agrupamientos.
RF 10	Visualizar agrupamientos.
RF 11	Importar datos de ejercicio de ordenamiento de tarjetas obtenidos por procedimientos tradicionales

Requerimientos No Funcionales

1. Usabilidad:
   1.1. El sistema podrá ser utilizado por cualquier persona que posea conocimientos básicos para interactuar con una computadora.
   1.2. Diseño sencillo, que propicie el desarrollo de habilidades en el uso del sistema sin mucha inversión de tiempo de entrenamiento.
2. Apariencia o interfaz externa:
   2.1. El sistema debe contar con las interfaces de usuario diseñadas de la manera más atractiva posible.
   2.2. Los colores que deben predominar en el sistema deben tener un tono claro evitando el uso de colores fuertes que perjudiquen la visión de los usuarios.
   2.3. Utilizará para los textos el tipo de fuente Arial con tamaño mínimo 8 y máximo 14. El color de los textos variará de acuerdo con el color de fondo.
   2.4. Los iconos de las tarjetas y categorías deben ser comprensibles de manera que especifiquen la funcionalidad que encierran cada uno de estos elementos.
   2.5. Debe predominar el uso de los íconos para la interacción con el usuario.
   2.6. El sistema debe contar con tres interfaces principales:
      2.6.1. Interfaz de preparación de ejercicios de ordenamiento de tarjetas.
      2.6.2. Interfaz de ejecución de ejercicios de ordenamiento de tarjetas.
      2.6.3. Interfaz de evaluación de ejercicios de ordenamiento de tarjetas.
3. Fiabilidad y disponibilidad:
   3.1. El sistema debe estar 100 % disponible, las 24 horas de un día, tanto para los Arquitectos de Información como a los miembros de los participantes, para ser utilizado.
   3.2. El sistema realizará una adecuada gestión de los errores que puedan ocurrir, garantizando la estabilidad e integridad del mismo.
4. Rendimiento:
   4.1. EL sistema debe crear cada tarjeta y categoría en no más de un segundo.

4.2. EL sistema debe crear un dendrograma[17] para cada algoritmo en no más de tres segundos.

5. Hardware:
   5.1. El sistema debe ejecutarse en un ordenador que tenga 256 MB de RAM como mínimo.
   5.2. El sistema debe ejecutarse en un ordenador que tenga 100 MB libres en el disco duro como mínimo.
6. Software:
   6.1. Máquina Virtual de Java versión 1.5 o superior.
7. Restricciones de diseño:
   7.1. El sistema será desarrollado en el lenguaje Java, sobre la plataforma de desarrollo J2SE 1.4 o superior.
8. Portabilidad:
   8.1. El sistema será portable. Al menos podrá ser ejecutado tanto en Microsoft Windows y GNU - Linux.
9. Ayuda y soporte técnico:
   9.1. El sistema debe contar con un manual de ayuda adjunto al mismo. Se deben exponer las principales funcionalidades del sistema así como habilitar búsqueda de información sobre cómo manejar el software.

Descripción del sistema propuesto

El sistema propuesto se divide en tres módulos lógicos: preparación, ejecución y evaluación de ejercicios de ordenamiento de tarjetas.

El módulo lógico de preparación de ejercicio de ordenamiento de tarjetas se encarga de todo lo relacionado con el diseño de los ejercicios de ordenamiento de tarjetas. Con este módulo lógico se pueden elaborar las tarjetas y las categorías necesarias para ejecutar un ejercicio de ordenamiento de tarjetas, así como definir el tipo del mismo, abierto o cerrado.

[17] Gráfico en forma de árbol.

El módulo de ejecución de ejercicios de ordenamiento de tarjetas permite que los participantes agrupen las tarjetas en clases haciendo uso de las categorías. Si el ejercicio de ordenamiento de tarjetas es abierto los participantes pueden crear nuevas categorías, en caso de que sea cerrado no. Este módulo permite almacenar el ejercicio de ordenamiento de tarjetas resuelto por cada participante.

Mediante el módulo lógico de evaluación de ejercicios de ordenamiento de tarjetas se puede realizar un análisis estadístico de las clases elaboradas por los participantes. Permite mezclar las soluciones realizadas por los participantes de un mismo ejercicio de ordenamiento de tarjetas, para su posterior análisis. Con este módulo se obtienen los clústeres luego de aplicar algoritmos de aglomerados jerárquicos y K – Means a dichos agrupamientos.

Actores del sistema

Actor	Descripción
Diseñador de ejercicios de ordenamiento de tarjetas.	Persona encargada de preparar un ejercicio de ordenamiento de tarjetas.
Participante.	Miembro de la audiencia del producto final seleccionados intencionadamente o al azar para participar en la ejecución de un ejercicio de ordenamiento de tarjetas.
Evaluador de ejercicios de ordenamiento de tarjetas.	Persona encargada de evaluar resultados obtenidos de la ejecución de un ejercicio de ordenamiento de tarjetas.

Modelo de Casos de Usos del Sistema

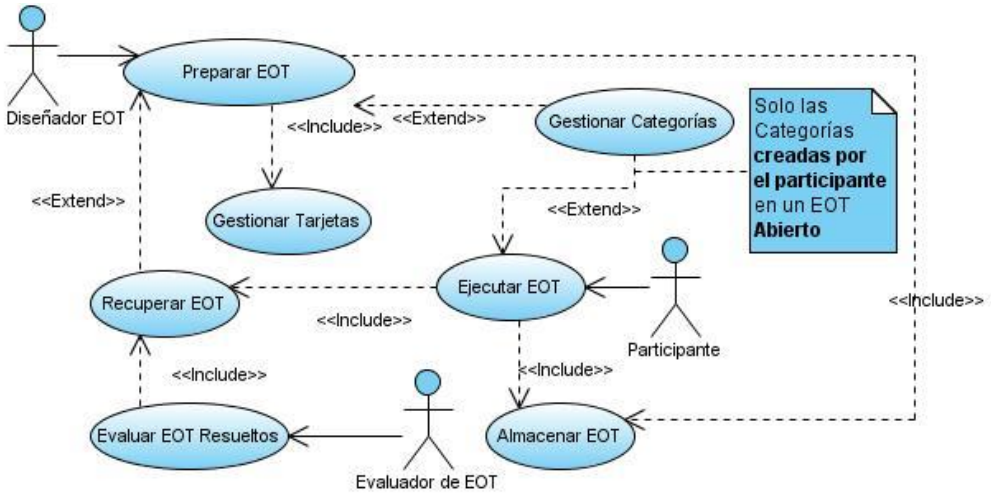


Fig. 6. Diagrama de Casos de Usos del Sistema

Caso de Uso del Sistema Preparar Ejercicio de Ordenamiento de Tarjetas

Caso de uso	
CU-1	Preparar EOT.
Propósito	Permite que un diseñador de EOT prepare un ejercicio de ordenamiento de tarjetas.
Actores	Diseñador de EOT.
Resumen	Se inicia cuando el diseñador de EOT solicita confeccionar un nuevo ejercicio de ordenamiento de tarjetas. El sistema le solicita la información necesaria para iniciar la preparación del ejercicio y le permite comenzar a confeccionar el EOT. Para lo cual debe elaborar las instrucciones, tarjetas y categorías.
Referencias	RF1, RF1.1 RF1.2, RF4, RF4.1, RF4.2, RF4.3, RF4.5, RF 4.4, RF 4.6.
Sección Crear EOT	
Flujo Básico	
Acción del actor	**Respuesta del sistema**
1 Selecciona crear un EOT.	1.1 Solicita el nombre, el tipo de EOT, y las instrucciones del EOT.
2 Introduce los datos solicitados.	2.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 2.2 Crea el EOT.
Flujos Alternos	
	1.1 Emite un mensaje de alerta sobre los campos incorrectos. 1.2 Regresa a la sección Crear EOT, a la transacción 1.1.
Sección Modificar EOT	
Flujo Básico	
Acción del actor	**Respuesta del sistema**
1 Selecciona abrir un EOT.	1.1 Ir al CUS Recuperar EOT en la sección Recuperar EOT. 1.2 Muestra el EOT.
2 Selecciona crear tarjeta.	2.1 Ir al CUS[18] Gestionar Tarjetas en la sección Crear Tarjeta.
3 Selecciona modificar tarjeta.	3.1 Ir al CUS Gestionar Tarjetas en la sección Modificar Tarjeta.
4 Selecciona eliminar tarjeta.	4.1 Ir al CUS Gestionar Tarjetas en la sección Eliminar Tarjeta.
5 Selecciona crear categoría.	5.1 Ir al CUS Gestionar Categorías en la sección Crear Categoría.
6 Selecciona modificar categoría.	6.1 Ir al CUS Gestionar Categorías en la sección Modificar Categoría.
7 Selecciona eliminar categoría.	7.1 Ir al CUS Gestionar Categorías en la sección Eliminar Categoría.
8 Selecciona guardar EOT.	8.1 Guarda los cambios en el fichero del EOT.
Flujos Alternos	
	1.1 Emite un mensaje de alerta sobre el error detectado. 1.2 Regresa a la sección Modificar EOT, a la transacción 1.2.
Puntos de Extensión	
Transacciones 6, 7, 8 de la sección Modificar EOT. Ver CUS Gestionar Categorías. Transacción 1 de la sección Modificar EOT. Ver CUS Recuperar EOT.	

Caso de Uso del Sistema Gestionar Tarjetas

Caso de uso	
CU-2	Gestionar Tarjetas.
Propósito	Permite que un diseñador de EOT gestione las tarjetas de un ejercicio de ordenamiento de tarjetas.
Actores	Diseñador de EOT.

[18] Caso de uso del sistema.

Resumen	Se inicia cuando el diseñador de EOT solicita gestionar las tarjetas de un ejercicio de ordenamiento de tarjetas. El sistema permite crear nuevas tarjetas, actualizar y visualizar la información contenida en las mismas, así como la eliminación de estas.
Referencias	RF4, RF4.1 RF4.2, RF 4.3.
Sección Crear Tarjeta	
Flujo Básico	
Acción del actor	**Respuesta del sistema**
1 Selecciona crear una tarjeta.	1.1 Solicita el nombre, y la descripción de la tarjeta.
2 Introduce los datos solicitados.	2.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 2.2 Crea y muestra la tarjeta.
Flujos Alternos	
	1.1 Emite un mensaje de alerta sobre los campos incorrectos. 1.2 Regresa a la sección Crear Tarjeta, a la transacción 1.1.
Sección Modificar Tarjeta	
Acción del actor	**Respuesta del sistema**
1 Elige la tarjeta a modificar.	1.1 Muestra el nombre, y la descripción actual de la tarjeta.
2 Modifica el nombre o la descripción de la tarjeta.	1.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 1.2 Modifica y muestra la tarjeta.
Flujos Alternos	
	1.1 Emite un mensaje de alerta sobre los campos incorrectos. 1.2 Regresa a la sección Modificar Tarjeta, a la transacción 1.1.
Sección Eliminar Tarjeta	
Acción del actor	**Respuesta del sistema**
1 Elige la tarjeta a eliminar.	1.1 Solicita confirmación de la acción.
2 Confirma la acción.	2.1 Elimina la tarjeta.

Caso de Uso del Sistema Gestionar Categorías

Caso de uso	
CU-3	Gestionar Categorías
Propósito	Permite que un diseñador de EOT o un participante de un EOT abierto, gestione las categorías de un ejercicio de ordenamiento de tarjetas.
Actores	Diseñador de EOT, participante.
Resumen	Se inicia cuando el diseñador de EOT o un participante de un EOT abierto, solicita gestionar las categorías de un ejercicio de ordenamiento de tarjetas. El sistema permite crear nuevas categorías, actualizar y visualizar la información contenida en las mismas, así como la eliminación de estas.
Referencias	RF4, RF4.4 RF4.5, RF4.6, RF5, RF5.1, RF5.1.1, RF5.1.2, RF5.1.3.
Sección Crear Categoría	
Flujo Básico	
Acción del actor	**Respuesta del sistema**
1 Elige crear una categoría.	1.1 Solicita el nombre, y la descripción de la categoría.
2 Introduce los datos solicitados.	2.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 2.2 Crea y muestra la categoría.
Flujos Alternos	
	1.1 Emite un mensaje de alerta sobre los campos incorrectos. 1.2 Regresa a la sección Crear Categoría, a la transacción 1.1.
Sección Modificar Categoría	
Flujo Básico	

Acción del actor	Respuesta del sistema
1 Elige la categoría a modificar.	1.1 Muestra el nombre, y la descripción actual de la categoría.
2 Modifica el nombre o la descripción de la categoría.	2.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 2.2 Modifica y muestra la categoría.
Flujos Alternos	
	1.1 Emite un mensaje de alerta sobre los campos incorrectos. 1.2 Regresa a la sección Modificar Categoría a la transacción 1.1.
Sección Eliminar Categoría	
Acción del actor	**Respuesta del sistema**
1 Elige la categoría a eliminar.	1.1 Solicita confirmación de la acción.
2 Confirma la acción.	2.1 Elimina la categoría.

Caso de Uso del Sistema Ejecutar Ejercicio de Ordenamiento de Tarjetas

Caso de uso	
CU-4	Ejecutar EOT.
Propósito	Permite que un participante ejecute un ejercicio de ordenamiento de tarjetas.
Actores	Participante.
Resumen	Se inicia cuando el participante solicita ejecutar un EOT. Para esto lee las instrucciones del EOT y agrupa las tarjetas sobre las categorías, a estos agrupamientos se le denominan clases. Si se ejecuta un EOT abierto el participante puede crear nuevas categorías.
Referencias	RF5, RF5.1, RF5.1.1, RF5.1.2, RF5.1.3, RF5.2, RF5.3, RF5.4, RF7.
Sección Ejecutar EOT	
Flujo Básicos	
Acción del actor	**Respuesta del sistema**
1 Selecciona ejecutar un EOT.	1.1 Ir al CUS Recuperar EOT en la sección Recuperar EOT.
	2.1 En caso de ser un EOT abierto, 2.1 ir al CUS Gestionar Categorías en las secciones Crear y Modificar Categorías, solo para las creadas por el participante.
3 Agrupa las tarjetas sobre las categorías.	3.1 Varía la composición de las clases.
4 Selecciona la tarjeta a desagrupar.	4.1 Varía la composición de las clases. 4.2 Retorna la tarjeta seleccionada a las tarjetas sin agrupar.
5 Solicita guardar los cambios realizados.	5.1 Ir al CUS Almacenar EOT en la sección Almacenar EOT.
Sección Modificar EOT Ejecutado	
Acción del actor	**Respuesta del sistema**
1 Selecciona modificar un EOT.	1.1 Ir al CUS Recuperar EOT en la sección Recuperar EOT. 1.2 Muestra las Soluciones existentes.
2 Elige la Solución a modificar.	2.1 Muestra la Solución seleccionada. 2.2 Ir a la sección Ejecutar EOT en la transacción 2.1.
Puntos de Extensión	
Transacción 3 sección Ejecutar EOT. Ver CUS Gestionar categorías.	

Caso de Uso del Sistema Evaluar Ejercicio de Ordenamiento de Tarjetas Resueltos

Caso de uso	
CU-5	Evaluar EOT Resueltos.
Propósito	Permite que el evaluador de EOT evalúe los resultados obtenidos de la ejecución de un ejercicio de ordenamiento de tarjetas.

Actores	Evaluador de EOT.
Resumen	Se inicia cuando el evaluador de EOT solicita evaluar la composición de las clases obtenidas de la ejecución de un EOT. El evaluador solicita al sistema que mezcle las diferentes soluciones de un mismo EOT, para su análisis. El evaluador indica los algoritmos mediante los cuales se realizará la evaluación y provee los parámetros necesarios para los mismos. El sistema muestra un informe con los resultados de la evaluación de las clases obtenidas de acuerdo con los algoritmos seleccionados y parámetros provistos.
Referencias	RF8, RF9, RF10, RF11.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
	1.1 Ir al CUS Recuperar EOT en la sección Recuperar EOT.
2 Si selecciona mezclar un EOT ir a la sección Mezclar EOT Ejecutados	
3 Si selecciona evaluar un EOT ir a la sección Evaluar EOT Ejecutados	
Sección Mezclar EOT Ejecutados	
Acción del actor	**Respuesta del sistema**
1 Selecciona mezclar un EOT.	1.1 Ir al CUS Recuperar EOT en la sección Recuperar EOT. 1.2 Para cada fichero de EOT ir al CUS Recuperar EOT en la sección Recuperar EOT. 1.3 Valida los EOT, en caso de que algún fichero de EOT seleccionado no sea del mismo EOT ir al flujo alterno 1.1. 1.4 Mezcla las soluciones de EOT. 1.5 Sobrescribe las soluciones en el fichero de EOT abierto originalmente con el conjunto de todas las soluciones de los ficheros seleccionados.
Flujos Alternos	
	1.1 Emite un mensaje de alerta indicando que los ficheros de EOT seleccionados no pueden ser mezclados.
Sección Evaluar EOT Ejecutados	
Acción del actor	**Respuesta del sistema**
1 Selecciona evaluar un EOT.	1.1 Solicita los parámetros de evaluación.
2 Introduce los datos solicitados.	2.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 2.3 Evalúa las soluciones del EOT. 2.4 Muestra los resultados.
Flujos Alternos	
	1.1 Emite un mensaje de alerta sobre los campos incorrectos. 1.2 Regresa a la sección Evaluar EOT, en la transacción 1.2.

Caso de Uso del Sistema Almacenar Ejercicio de ordenamiento de tarjetas

Caso de uso	
CU-6	Almacenar EOT.
Propósito	Garantizar la persistencia de un EOT, en sus diferentes estados.
Actores	Diseñador de EOT, participantes, evaluador de EOT.
Resumen	Se inicia cuando el actor selecciona almacenar un EOT. El sistema solicita los datos necesarios y procede a almacenar el EOT.
Referencias	RF3, RF6.
Sección Almacenar EOT	
Flujo Básico	
Acción del actor	**Respuesta del sistema**

1 Selecciona guardar un EOT.	1.1 Solicita la ubicación y el formato del fichero del EOT.
2 Introduce los datos solicitados.	2.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 2.2 Almacena el EOT en el fichero.
Flujos Alternos	
	1.3 Emite un mensaje de alerta sobre el error detectado. 1.4 Regresa a la sección Almacenar EOT, a la transacción 1.1.

Caso de Uso del Sistema Recuperar Ejercicio de ordenamiento de tarjetas

Caso de uso	
CU-7	Recuperar EOT.
Propósito	Recupera de un EOT, en sus diferentes estados.
Actores	Diseñador de EOT, participantes, evaluador de EOT.
Resumen	Se inicia cuando el actor selecciona abrir un EOT. El sistema solicita los datos necesarios y procede a cargar el EOT.
Referencias	RF2.
Sección Recuperar EOT	
Flujo Básico	
Acción del actor	**Respuesta del sistema**
1 Selecciona abrir un EOT.	1.1 Solicita la ubicación del fichero del EOT.
2 Introduce los datos solicitados.	2.1 Valida los datos obtenidos, en caso de que no estén correctos ir al flujo alterno 1.1. 2.2 Carga y muestra el EOT.
Flujos Alternos	
	1.5 Emite un mensaje de alerta sobre el error detectado. 1.6 Regresa a la sección Modificar EOT, a la transacción 1.1.

Conclusiones del capítulo

El sistema a implementar abarcará las fases de preparación, ejecución y evaluación del ordenamiento de tarjetas. El mismo dispondrá del algoritmo de formación de aglomerados planos K – Means y de los algoritmos de formación de aglomerados jerárquicos Single – Link, Complete – Link, Group – Average y Centroid – Link. Asimismo, el análisis realizado a los requisitos identificados evidencia que el sistema reunirá las condiciones para lograr un equilibrio entre usabilidad, funcionalidad y disponibilidad.

Capítulo 3. Construcción del sistema

Existe una serie de principios y buenas prácticas que a lo largo de los años de experiencia en la elaboración de arquitecturas de software han ayudado considerablemente a los ingenieros informáticos ante problemas comunes de organización y flujo de datos entre componentes de un sistema informático. Son comúnmente llamados patrones de arquitectura.

Un patrón es una solución a un problema en un contexto, codifica conocimiento específico acumulado por la experiencia en un dominio. Describe un problema que se instancia varias veces en un dominio determinado y propone la solución a ese problema. En el caso de los patrones arquitectónicos proponen un esquema organizativo estructural para los sistemas informáticos.

Patrón Modelo-Vista-Controlador

El patrón arquitectónico Modelo Vista Controlador (en lo adelante MVC) separa los datos de una aplicación, la lógica de la misma y las interfaces de usuario en tres estructuras distintas las cuales tienen definidas reglas de comunicación. El Modelo es referido a la representación de la información con la cual un sistema informático opera. La Vista se encarga de visualizar de una manera entendible los datos obtenidos desde el modelo. El controlador responde ante los eventos de los usuarios de acuerdo con sus reglas de gestión de eventos definidas, pudiendo dirigir cambios en el Modelo. Adoptar MVC como patrón de arquitectura proporciona una buena organización del código generado, su reutilización y flexibilidad, ofrece un punto de entrada único para toda la aplicación y reduce el tiempo necesario para modificar y añadir mejoras a una aplicación. Es muy apropiado para aplicaciones Web.

Patrón arquitectónico de Tres Capas

Una arquitectura de software diseñada en capas consiste en la definición de niveles de abstracción, los cuales tienen una función específica permitiendo un diseño modular. Ello permite que se creen sistemas con un bajo acoplamiento entre sus módulos o componentes. Una variante de este patrón muy utilizada es la de Tres Capas. Mediante la cual se fracciona al sistema informático en la capa de Presentación, la capa de Lógica del Negocio y la capa de Acceso a Datos.

“La capa de Presentación es la que se encarga de interactuar con el usuario mediante la interfaz de usuario. La capa Lógica del Negocio, llamada también como Lógica de la Aplicación, se encarga de realizar las tareas para las cuales está concebido el sistema. Es implementada utilizando un modelo orientado a objetos del dominio de la aplicación. Se encarga de controlar las operaciones de acuerdo con las reglas del negocio. La capa de Acceso a Datos es la que gestiona el almacenamiento de los datos, ya sea en una base de datos o en un fichero, así como la consulta a los mismos” (Flower, 2003).

Una restricción de este patrón consiste en que las capas inferiores no deben de conocer ni hacer llamadas a procedimientos implementados en capas superiores, sino que las funcionalidades que ellas ofrecen son accedidas desde niveles mayores (Larman, 1999).

Para el desarrollo de la solución se adoptará una arquitectura de tres capas debido sus potencialidades para la reutilización y el aprovechamiento de los beneficios de una clara separación entre las funciones desplegadas en capas. Además, permite la estandarización y proporciona una distribución clara del trabajo entre los miembros de un equipo de desarrollo.

Patrones GRASP

Los patrones GRASP (General Responsibility Assignment Software Patterns, en español Patrones Generales de Software para la Asignación de Responsabilidades) describen los principios fundamentales de la asignación de responsabilidades a objetos (Larman, 1999).

En el desarrollo de esta solución se aplicaron los patrones GRASP: Experto, Creador, Controlador, No Hables con Extraños, Bajo Acoplamiento, Alta Cohesión, Fabricación Pura, Indirección y Polimorfismo.

El patrón Experto soluciona el problema de asignar una responsabilidad de forma general, al recomendar para esta función a la clase que maneje la información necesaria. El patrón Creador trata el problema de qué clase debe ser la encargada de instanciar a otra. El patrón Controlador ayuda a decidir quién se encarga de administrar un evento del sistema. El patrón Bajo Acoplamiento recomienda asignar las responsabilidades de tal forma que se le dé soporte a una mayor reutilización y poca dependencia. El patrón Alta Cohesión aconseja mantener la clase lo más cohesionada posible para controlar la

complejidad de la misma. El patrón Fabricación Pura propone crear una clase artificial fuertemente cohesionada que no represente nada en el dominio para dar soporte a una alta cohesión, un bajo acoplamiento y a la reutilización. El patrón Indirección plantea asignar la responsabilidad a un objeto intermedio para que medie entre otros componentes o servicios de modo que no se acoplen directamente. El patrón No Hables con Extraños ayuda a asignar las responsabilidades de modo que se desconozca la estructura de los objetos indirectos. El patrón Polimorfismo recomienda la definición de un método homónimo en la clase generalización y en cada subclase con un comportamiento específico (Larman, 1999).

Patrones GoF

Los patrones Singleton y Fachada son patrones de diseño creado por “La Pandilla de los Cuatro” (Gang of Four, GoF). Mediante el patrón Singleton se garantiza que una clase solo pueda ser instanciada una sola vez. El patrón Fachada propone crear una clase que unifique las funciones de un conjunto de clases y que esta sea la encargada de colaborar con el sistema (Larman, 1999).

Modelo de Diseño

Para una mejor comprensión de la solución y de acuerdo con la arquitectura de tres capas se han organizado las clases en tres paquetes principales: Presentación, Lógica del Negocio y Acceso a Datos. Cada uno de los paquetes se ha dividido a la vez en sub-paquetes de acuerdo con las funcionalidades de las clases contenidas (Fig. 10). Cada paquete representa una capa de la arquitectura y está construido sobre su predecesor. Las clases pertenecientes a una capa están relacionadas con las clases de la capa inmediata inferior y desconocen a las clases de las capas superiores.

Acceso a Datos

La capa de Acceso a Datos es la encargada de la persistencia y recuperación de objetos, específicamente de la interacción de la aplicación con los ficheros de almacenamiento de ejercicios de ordenamiento de tarjetas (EOT). Estos ficheros pueden contener EOT en forma de objeto serializado o de documento XML. El paquete de Acceso a Datos mantiene un bajo acoplamiento con las entidades del negocio. Las clases de este paquete desconocen a las entidades del negocio y solo se centran en la entrada y salida de datos.

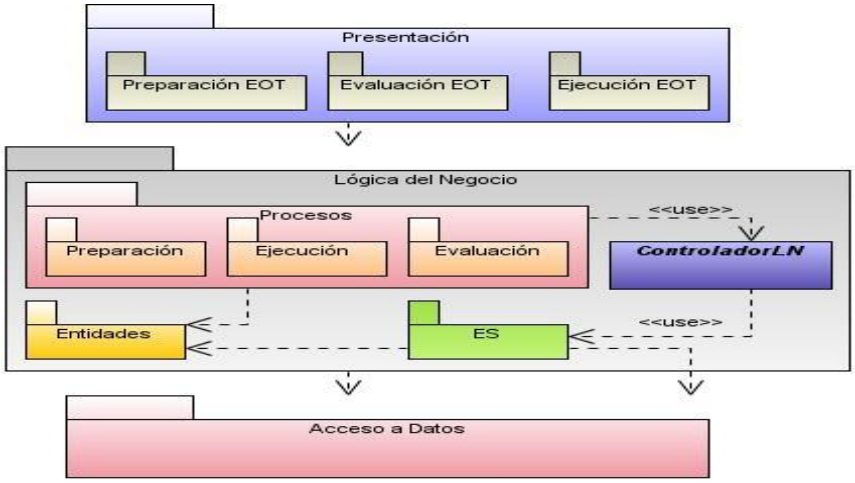


Fig. 7. Diagrama de paquetes del sistema

Este paquete cuenta con cuatro clases: ControladorES, Escritor, Lector y ArchivoEOT. Esta última representa el fichero físico en el que se encuentra almacenada la información a recuperar o modificar. La información es recuperada mediante el método leer de la clase Lector, este método retorna un objeto genérico que será transformado en la capa superior. Para modificar o crear fichero se utiliza el método escribir de la clase Escritor, este método recibe un objeto genérico el cual es plasmado en un fichero serializado o XML. La clase ControladorES es la encargada de servir de fachada a la capa para el acceso desde la capa superior. Además, contiene una interfaz: IESPaq, que es la encargada de garantizar la comunicación desde la capa superior.

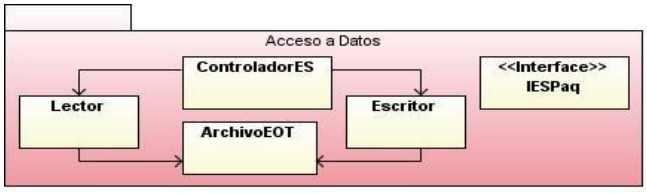


Fig. 8. Capa de Acceso a Datos

Nombre: ControladorES	
Almacenar	
Nombre:	escribir(eot: Object, camino: string)
Descripción:	Garantiza la persistencia de un EOT.
Recuperar	
Nombre:	leer(): Object
Descripción:	Garantiza la recuperación de un EOT.

Lógica del Negocio

La capa Lógica del Negocio engloba a las clases encargadas de ejecutar las tareas y reglas que rigen el proceso (Soulie, 2009). En el diseño orientado a objetos la capa Lógica del Negocio se divide en otras menos densas. En esta solución el paquete de Lógica del Negocio está dividida en tres paquetes: Procesos, Entidades y Entrada Salida (ES).

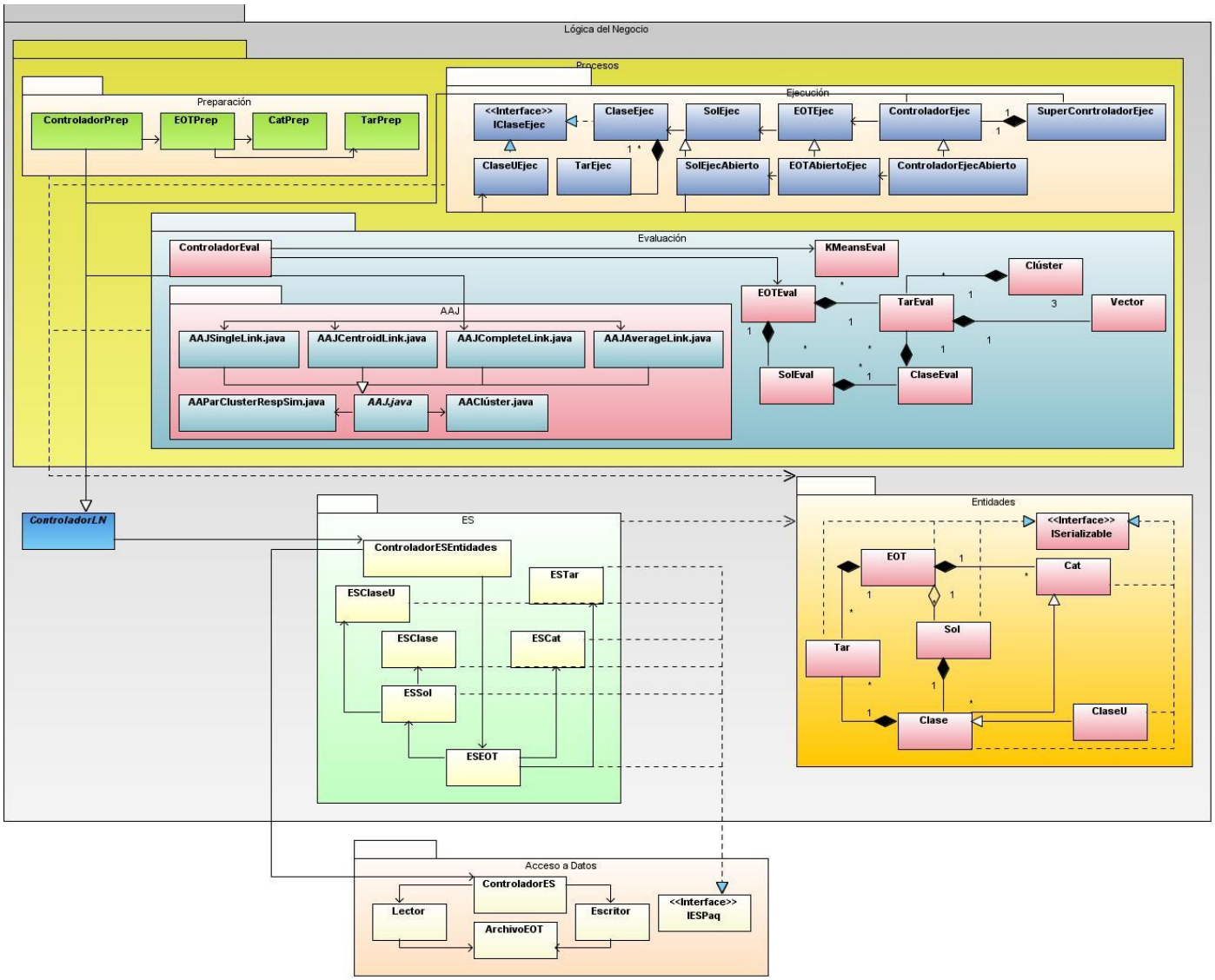


Fig. 9. Paquete Lógica del Negocio y su vínculo con el Paquete de Acceso a Datos.

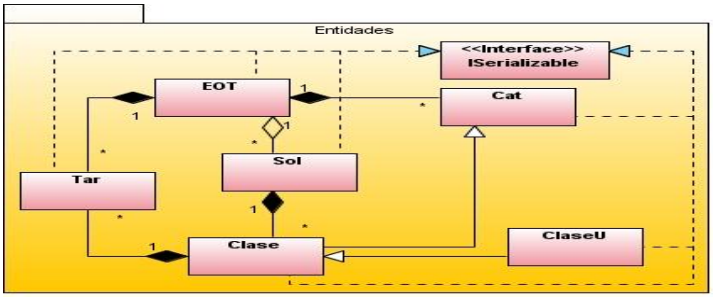


Fig. 10. Paquete Entidades.

El paquete Entidades es la base de la capa Lógica del Negocio. Las clases de este paquete son las encargadas de almacenar la información básica de las entidades. El paquete ES garantiza la comunicación hacia la capa de Acceso a Datos, para esto implementa la interfaz de la capa de Acceso a Datos IESPaq. El paquete Procesos contiene los paquetes Preparación, Ejecución y Evaluación. Cada paquete cuenta con una especialización de las clases contenidas en el paquete Entidades acorde a las necesidades del mismo. Cada paquete cuenta con un controlador que funge como punto de contacto con la capa superior.

Nombre: ControladorLN	
Atributo	
Eot	EOT
Recuperar EOT	
Nombre:	leer(camino: string)
Descripción:	Garantiza la recuperación de un EOT.
Guardar EOT	
Nombre:	guardar(camino: string)
Descripción:	Garantiza la persistencia de un EOT.

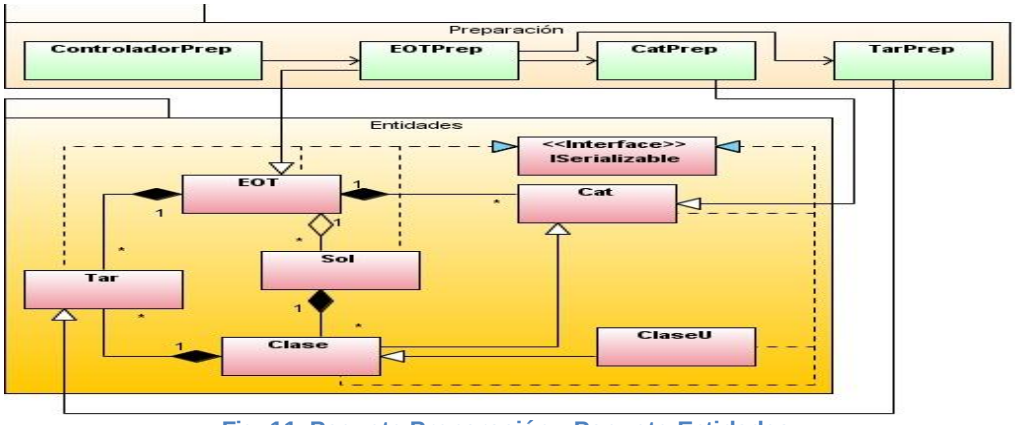


Fig. 11. Paquete Preparación y Paquete Entidades.

El paquete Preparación engloba las clases responsables de la preparación de un EOT. Contiene la clase ControladorPrep que sirve de fachada para el paquete.

Nombre: ControladorPrep	
Atributo	
Eot	EOTPrep
Recuperar EOT	
Nombre:	leer(camino: string)
Descripción:	Garantiza la recuperación de un EOTPrep.
Crear EOT	
Nombre:	crearEOT(nombre: String, instrucc: String, bierto: boolean)
Descripción:	Crea un EOT.
Modificar instrucción del EOT	
Nombre:	modificarInst(inst: String)
Descripción:	Modifica la instrucción del EOT.
Agregar Tarjeta	
Nombre:	agregarTar(nombre: String, descript: String): int
Descripción:	Agrega una nueva tarjeta a un EOT.
Modificar Tarjeta	
Nombre:	modificarTar(idTar: int, nombre: String, descript: String)
Descripción:	Modifica una tarjeta de un EOT.
Eliminar Tarjeta	
Nombre:	eliminarTarjeta(id:int)
Descripción:	Elimina una tarjeta de un EOT.
Agregar categoría	
Nombre:	agregarCat(nombre: String, descript: String): int
Descripción:	Agrega una nueva categoría a un EOT.
Modificar categoría	
Nombre:	modificarCategoría(idCat: int, nombre: String, descripción: String)
Descripción:	Modifica una categoría de un EOT.
Eliminar categoría	
Nombre:	eliminarCategoría(idCat: int)
Descripción:	Elimina una categoría de un EOT.

El paquete Ejecución contiene las clases responsables de la ejecución de un EOT abierto o cerrado. Este paquete a diferencia de los demás, cuenta con tres controladores: SuperControladorEjec, ControladorEjec y ControladorEjecAbierto. El controlador SuperControladorEjec es el encargado de leer el EOT a ejecutar, determinar si es abierto o cerrado y en base a esto generar el controlador correspondiente: ControladorEjec o ControladorEjecAbierto. Los controladores ControladorEjec, y ControladorEjecAbierto se encargan de la ejecución del ejercicio de acuerdo con el tipo de EOT.

Nombre: ControladorEjec	
Atributo	
Eot	EOTEjec
Agregar Solución	

Nombre:	agregarSolución(): int
Descripción:	Genera una nueva Solución de un EOT.
Eliminar Solución	
Nombre:	desagregarSol(idSol: int)
Descripción:	Elimina una Solución de un EOT.
Agrupar Tarjeta	
Nombre:	agregarTarjetaAClase(idSol: int, idClase: int, idTar: int)
Descripción:	Establece la relación entre una clase y una tarjeta.
Desagrupar Tarjeta	
Nombre:	desagruparTarjeta(idSol: int, idTar: int)
Descripción:	Rompe la relación entre una clase y una tarjeta

Nombre: ControladorEjecAbierto	
Atributo	
Eot	EOTEjecAbierto
Agregar clase	
Nombre:	agregarClaseU(idSol: int, nombre: String, descript: String): int
Descripción:	Agrega una nueva clase creada por el participante.
Eliminar clase	
Nombre:	eliminarClaseU(idSol: int, idClase: int)
Descripción:	Elimina una clase creada por el participante.
Modificar el nombre de una clase agregada por el participante	
Nombre:	modificarNombreClaseU(idSol: int, idClase: int, nombre: String)
Descripción:	Modifica el nombre de una clase agregada por el participante.
Modificar la descripción de una clase agregada por el participante	
Nombre:	modificarDescripClaseU (idSol: int, idClase: int, descrip: String)
Descripción:	Modifica la descripción de una clase agregada por el participante.

Nombre: SuperControladorEjec	
Recuperar EOT	
Nombre:	leer(camino: String)
Descripción:	Recupera un EOT, y establecer el tipo de EOT en cuestión.

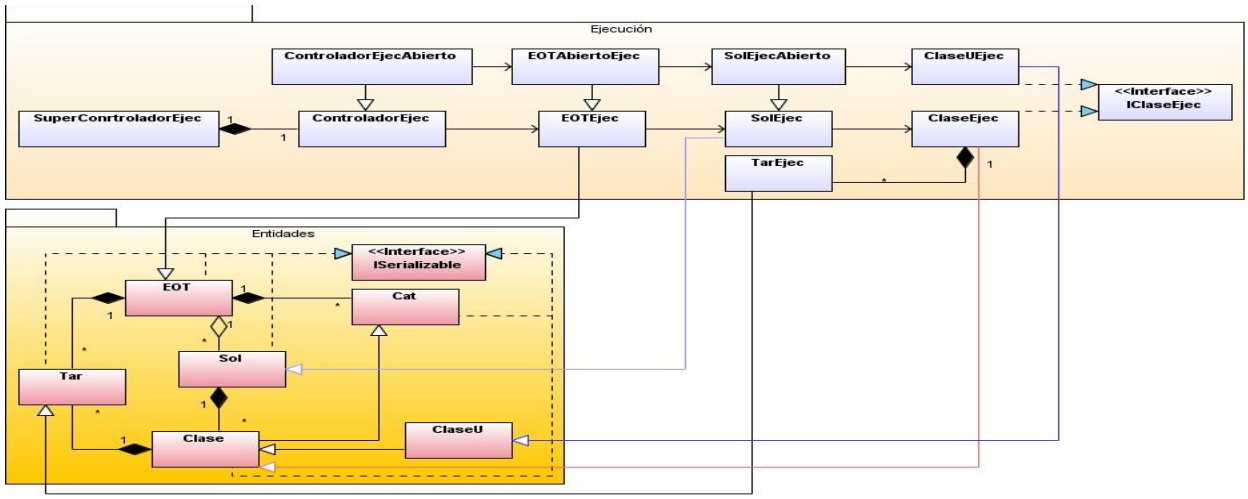


Fig. 12. Paquete Ejecución.

El paquete Evaluación contiene las clases encargadas de evaluar los EOT mediante el algoritmo K – Means y el sub – paquete AAJ (Analizador de Aglomerados Jerárquicos), el cual engloba las clases que realizan el análisis de aglomerados jerárquicos. A través de la clase ControladorEval se puede solicitar la evaluación de un EOT mediante los cuatro algoritmos de aglomerados jerárquicos implementados.

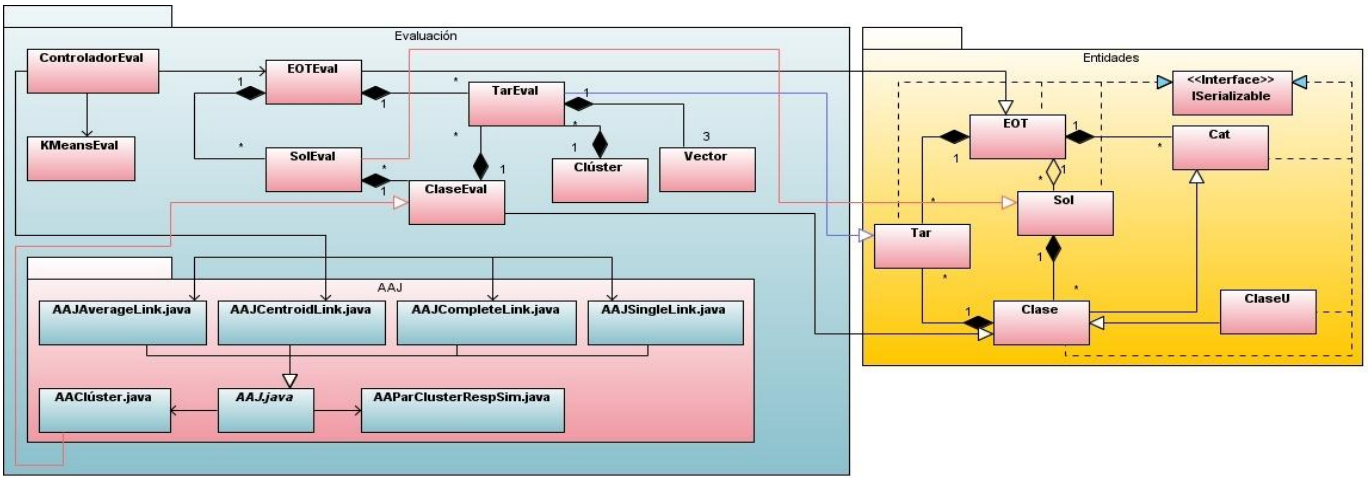


Fig. 13. Paquete Evaluación y Paquete Entidades.

Nombre: ControladorEval	
Recuperar EOT	
Nombre:	leer(camino: string)
Descripción:	Garantiza la recuperación de un EOTEval.
Analizar con K-Means	
Nombre:	evluarKMeans(cantidadDeClústeres: int, cantidadDelter: int, RSS: Double, varRSS: Double): ArrayList<Clúster>
Descripción:	Evalúa un EOT mediante K – Means.
Analizar con Centroid Link	
Nombre:	evaluarAAJCentroidLink(): AAClúster
Descripción:	Evalúa un EOT mediante Centroid Link.
Analizar con Complete Link	
Nombre:	evaluarAAJCompleteLink():AAClúster
Descripción:	Evalúa un EOT mediante Complete Link.
Analizar con Single Link	
Nombre:	evaluarAAJSingleLink():AAClúster
Descripción:	Evalúa un EOT mediante Single Link.

Capa Presentación

La capa de Presentación es la encargada de lograr la comunicación directa entre el sistema y el usuario final a través de una interfaz gráfica. Una interfaz gráfica de usuario consiste en un conjunto de componentes empleados por usuarios para comunicarse con los sistemas informáticos. Los usuarios

dirigen el funcionamiento del sistema mediante la generación de eventos.

Para una mayor organización de la capa Presentación queda dividida en paquetes según las funcionalidades que brinde cada uno. Los paquetes empleados son PreparaciónEOT, EjecuciónEOT, EvaluaciónEOT y el paquete Elementos Genéricos.

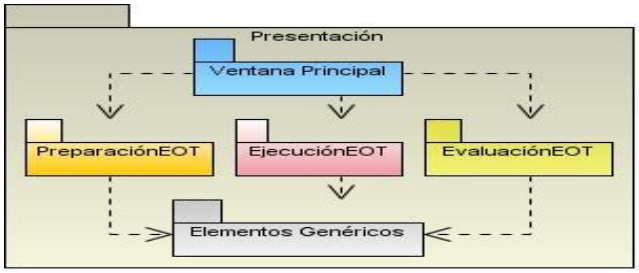


Fig. 14. Capa Presentación.

El paquete Ventana Principal agrupa la interfaz gráfica principal que actúa directamente el usuario final del sistema así como su clase controladora de eventos ControladorVisualGeneral. Dicha clase tiene la responsabilidad de gestionar todos los eventos visuales generados en la interfaz gráfica principal y delegar las responsabilidades al paquete correspondiente para que realice el procesamiento del evento a través de su controlador.

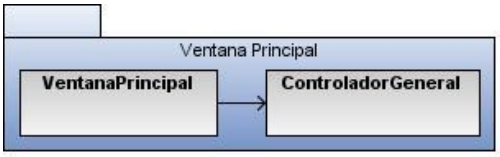


Fig. 15. Paquete Ventana Principal

Nombre: ControladorGeneral	
Atributo	**Tipo**
ventanaPrincipal	VentanaPrincipal
controladorAnalisis	ControladorVisualAnalisis
controladorPreparación	ControladorVisualPreparación
controladorSolución	ControladorVisualEjecución
Instancia	static ControladorVisualGral
Adicionar Tablero Kmeans	
Nombre:	AdicionarTableroKmeans()
Descripción:	Permite gestionar la confección de un tablero que muestre el resultado del algoritmo KMeans.
Solicitar nuevo EOT	
Nombre:	**solicitarNuevoEOT**

Descripción:	Solicita que sea confeccionado un nuevo EOT
Solicitar analizar resultado	
Nombre:	solicitarAnalizarResultado()
Descripción:	Solicita que sean analizados las soluciones de los EOT importados
Solicitar resolver un EOT	
Nombre:	solicitarResolverEjercicio()
Descripción:	Solicita que un EOT sea resuelto
Solicitar abrir EOT	
Nombre:	solicitarAbrirEOT()
Descripción:	Solicita que un EOT sea abierto
Solicitar guardar EOT	
Nombre:	solicitarGuardarEOT()
Descripción:	Solicita que un EOT sea guardado

El paquete Elementos Genéricos agrupa los elementos visuales comunes con los cuales se construirán los demás componentes del sistema. Por su condición de abstracción ninguna de las clases que encapsula el paquete puede ser instanciadas en la capa. De las mismas heredarán las clases que en realidad desempeñen el papel de interacción entre usuario final y sistema.

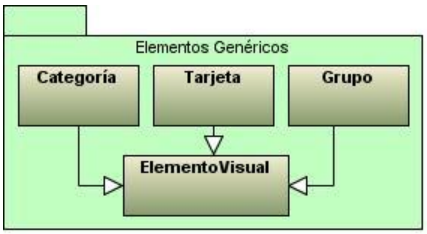


Fig. 16. Paquete Elementos Genéricos

El paquete PreparaciónEOT se encarga de la gestión de los eventos visuales para la creación de un EOT. La clase ControladorPreparación es la encargada de controlar toda la preparación de un EOT, la clase ContenedorPreparación contiene los diferentes tableros (TableroTarjetaPreparación y TableroCategoríaPreparación), que a su vez están compuestos por instancias de TarjetaPreparación y CategoríaPreparación.

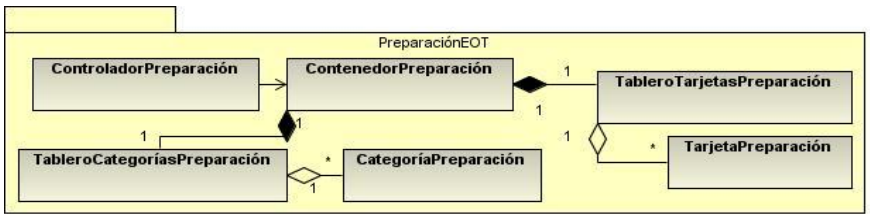


Fig. 17 Paquete PreparaciónEOT.

Nombre: ControladorPreparación	
Atributo	**Tipo**
ventanaPrincipal	VentanaPrincipal
Contenedor	ContenedorPreparación
Propiedades	PropiedadesElemento
Instancia	ControladorVisualPreparación
controladorLogicoPrep	ControladorPrep
Ejercicio	EOTPrep
Crear nuevo EOT	
Nombre:	nuevoEjercicio():void
Descripción:	Permite confeccionar la preparación de un EOT
Abrir EOT	
Nombre	abrirEjercicio()
Descripción	Solicita abrir un EOT
Solicitar guardar	
Nombre	solicitarGuardar ()
Descripción	Solicita guardar un EOT.
Cargar EOT	
Nombre	cargarEjercicio()
Descripción	Carga un EOT abierto
Solicitar modificar una tarjeta de Preparación	
Nombre:	modificarTarjetaPreparación(nombre: string, descripción: string, incluirDescripción:boolean)
Descripción:	Ordena que se modifica una tarjeta de Preparación con los nuevos valores indicados
Solicitar adicionar una tarjeta de Preparación	
Nombre:	adicionarTarjetaPreparación(nombre: string, descripción: string, incluirDescripción:boolean)
Descripción:	Ordena que se adicione una nueva tarjeta de Preparación a la preparación del EOT
Eliminar una tarjeta de Preparación	
Nombre:	eliminarTarjetaPreparación()
Descripción:	Solicita que se elimine una tarjeta de Preparación
Solicitar modificar una categoría	
Nombre:	modificarCategoría(nombre: string, descripción: string, incluirDescripción:boolean)
Descripción:	Ordena que se modifica una categoría con los nuevos valores indicados
Solicitar adicionar una categoría	
Nombre:	adicionarCategorla(nombre: string, descripción: string, incluirDescripción:boolean)
Descripción:	Ordena que se adicione una nueva categoría a la preparación del EOT
Eliminar una categoría	
Nombre:	eliminarCategoría()
Descripción:	Solicita que se elimine una categoría

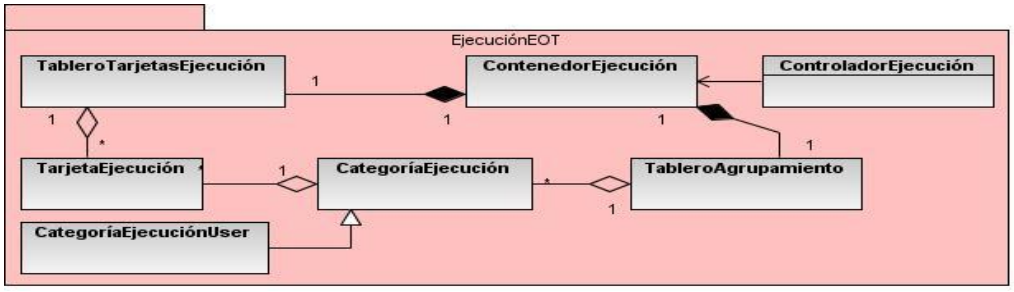


Fig. 18. Paquete EjecuciónEOT

El paquete EjecuciónEOT se encarga de la gestión de los eventos visuales para la ejecución de un EOT.

La clase ControladorVisualEjecución controla los eventos visuales en la ejecución de un EOT; la clase ContenedorEjecución contiene los tableros contenedores de categorías y tarjetas para el agrupamiento.

Nombre: ControladorEjecución	
Atributo	**Tipo**
controlador	ControladorEjecución
ventanaPrincipal	VentanaPrincipal
ContenedorEjecución	ContenedorEjecución
controladorLogicoEjec	ControladorEjec
Ejercicio	EOTEjec
ejercicioAbierto	EOTAbiertoEjec
Solicitar Agrupar	
Nombre:	solicitarAgrupar()
Descripción:	Permite iniciar el proceso de agrupamiento.
Solicitar desagrupar	
Nombre:	solicitarDesagrupar()
Descripción:	Permite iniciar el proceso de desagrupamiento de una tarjeta de Ejecución
Solicitar modificar EOT	
Nombre	solicitarModificarEOTEjecutado()
Descripción	Permite modificar una ejecución de un EOT
Cargar Ejercicio	
Nombre:	cargarEjercicio(ejercicioPreparado: EOTPrep)
Descripción:	Carga los datos de un EOT preparado

El paquete EvaluaciónEOT agrupa las clases encargadas de la gestión de los eventos visuales para la evaluación de un EOT. Los eventos visuales son controlados mediante la clase ControladorEvaluación además de garantizar el acoplamiento con la clase controladora de la Capa LógicaDelNegocio. La clase ContenedorEvaluación agrupa los distintos gráficos responsables de mostrar los resultados procesados por los algoritmos. Estos gráficos son visualizados a través de las clases ContenedorSingleLink, TableroKmeans, ContenedorGALink y ContenedorCompleteLink.

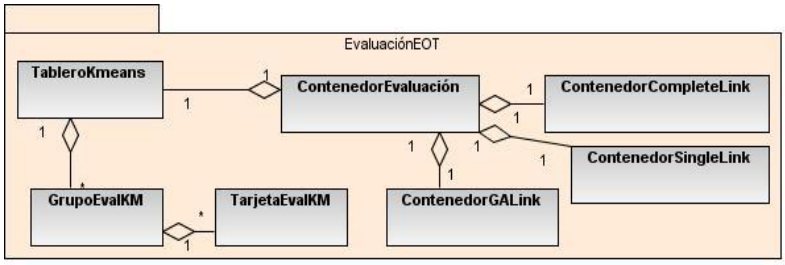


Fig. 19. Paquete EvaluaciónEOT

Nombre: ControladorEvaluación

Atributo		**Tipo**
instancia		ControladorAnalisis
contenedorCompleteLink		ContenedorCompleteLink
contenedorSingleLink		ContenedorSingleLink
contenedorGALink		ContenedorGALink
tableroKmeans		TableroKMeans
controladorLogicoEvaluación		ControladorEval
ejercicioLogicoEOTEval		EOTEval
Solicitar analizar resultado		
Nombre:	solicitarAnalizarResultado()	
Descripción:	Solicita al análisis de los resultados	
Analizar vía KMeans		
Nombre:	analizarKMeans()	
Descripción:	Genera los resultados por el algoritmo KMeans	
Importar Solución		
Nombre:	importarSolucion()	
Descripción:	Permite que sea importada una solución	
Importar Soluciones		
Nombre:	importarSoluciones()	
Descripción:	Permite que sean importadas varias soluciones	
Abrir Evaluación		
Nombre:	abrirEOT():boolean	
Descripción:	Permite que un EOT sea abierto para ser evaluado	
Llenar datos manualmente		
Nombre:	llenarDatosManuales():	
Descripción:	Permite que el arquitecto de información introduzca las ocurrencias entre las tarjetas	

Conclusiones del capítulo

Adoptar una arquitectura de tres capas para el desarrollo de la solución permitió crear clases más cohesionadas, con lo cual se potencia el bajo acoplamiento y la reutilización de las mismas. El desacoplamiento de las entidades del negocio y de la capa de Acceso a Datos permite que esta pueda ser reutilizada en el desarrollo de otras herramientas de la plataforma ABAD. El diseño de la capa de Presentación permite la incorporación de nuevos componentes para representar los resultados de la técnica de ordenamiento de tarjetas. Por su parte, el diseño de la capa Lógica del Negocio permite su escalabilidad mediante la reutilización de las clases que representan las entidades del negocio. La abstracción de las entidades del negocio, de los procesos en los que están involucradas, potencia la reutilización y la escalabilidad del sistema en su conjunto.

Capítulo 4. Implementación y prueba

Estándares de Codificación

Un estilo o estándar de codificación es el conjunto de reglas o normas usadas para escribir código y que incluye una gran gama de aspectos dentro del proceso de codificación (UCI, 2010). El uso de un estilo de codificación contribuye a la comunicación dentro del equipo de desarrollo, al mantenimiento del software, y a la reducción de errores.

La notación Camel consiste en escribir los identificadores con la primera letra de cada palabra en mayúsculas y el resto en minúscula: EndOfFile. Se denomina notación "Camel" porque los identificadores recuerdan las jorobas de un camello. Tiene dos variantes (UCI, 2010):

- UpperCamelCase, CamelCase o PascalCase: La primera letra también es mayúscula.
- lowerCamelCase, camelCase o dromedaryCase: La primera letra es minúscula.

Teniendo en cuenta que la solución se desarrollará en Java se decidió utilizar la notación CamelCase en identificadores de clases, y dormedaryCase para métodos y variables, siguiendo las recomendaciones de notación para este lenguaje de programación (UCI, 2010). Para la sangría se determinó utilizar el estilo BSD[19], debido a las ventajas de visibilidad que el mismo ofrece (Fig. 20).

```
public boolean igualA(Vector otro)
{
    if(this.arreglo.size() != otro.arreglo.size())
    {
        return false;
    }
    for (int i = 0; i < this.arreglo.size(); i++)
    {
        if (this.arreglo.get(i) != otro.arreglo.get(i))
        {
            return false;
        }
    }
    return true;
}
```

Fig. 20. Segmento de código con el estilo aplicado.

[19] Berkeley Software Distribution.

XML

El Lenguaje de Marcas Extensible (XML, Extensible Markup Language) es una versión del Lenguaje de Marcas Estándar Generalizado (ISO 8879) (SGML, Standard Generalized Markup Language). Un lenguaje de marcas es un conjunto de comandos que le indica a un programa cómo mostrar el contenido. Dichos comandos están escritos en texto sencillo para que sean inteligibles para las personas. Los esquemas XML definen los elementos y atributos presentes en un documento XML, son la base de la organización de la información del mismo (Mercer, 2001). La solución debe recuperar información de ficheros XML, por lo que es necesario un esquema para garantizar la organización de la información en el archivo. A continuación se expone el esquema XML propuesto:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <xsd:complexType name="EOT">
        <xsd:element name="tarjeta" type="tarjetaType" />
        <xsd:element name="categoria" type="categoria" />
        <xsd:element name="solucion" type="solucionType" />
        <xsd:attribute name="id" type="xsd:integer"/>
        <xsd:attribute name="nombre" type="xsd:string" />
        <xsd:attribute name="instrucciones" type="xsd:string" />
        <xsd:attribute name="abierto" type="xsd:boolean" />
        <xsd:attribute name="fecha" type="xsd:date" />
   </xsd:complexType>
   <xsd:complexType name="tarjetaType">
      <xsd:attribute name="id" type="xsd:integer" />
      <xsd:attribute name="nombre" type="xsd:string" />
      <xsd:attribute name="descripcion" type="xsd:string" />
   </xsd:complexType>
   <xsd:complexType name="categoriaType">
      <xsd:attribute name="id" type="xsd:integer" />
      <xsd:attribute name="nombre" type="xsd:string" />
      <xsd:attribute name="descripcion" type="xsd:string" />
   </xsd:complexType>
   <xsd:complexType name="claseType">
       <xsd:sequence>
           <xsd:element name="tarjeta" type="tarjetaType" minOccurs="1" maxOccurs="unbounded"/>
       </xsd:sequence>
      <xsd:attribute name="id" type="xsd:integer"/>
      <xsd:attribute name="nombre" type="xsd:string" />
      <xsd:attribute name="descripcion" type="xsd:string" />
   </xsd:complexType>
   <xsd:complexType name="solucionType">
       <xsd:sequence>
           <xsd:element name="clase" type="claseType" minOccurs="1" maxOccurs="unbounded"/>
       </xsd:sequence>
       <xsd:attribute name="id" type="xsd:integer" />
       <xsd:attribute name="fecha" type="xsd:date" />
   </xsd:complexType>
   <xsd:element name="eot" type="EOT" />
</xsd:schema>
```

Fig. 21. Esquema XML utilizado en la solución

Modelo de Despliegue

“El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo” (Jacobson, et al., 2000).La presente solución es una aplicación para entornos de escritorio por lo que será ejecutada en una estación de trabajo, independientemente de que la misma disponga de conexión de red o no. A continuación se muestra el modelo de despliegue de la solución (Fig. 22).

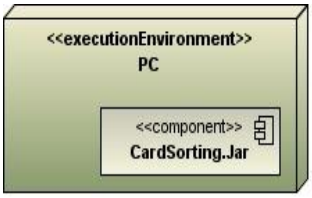


Fig. 22. Modelo de Despliegue

Modelo de Implementación

“El modelo de implementación describe cómo los elementos del modelo de diseño, como las clases, se implementan en términos de componentes, como ficheros de código fuente, ejecutables, etc. El modelo de implementación describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y cómo dependen unos de otros” (Jacobson, et al., 2000).

Para la presente solución se implementaron tres subsistemas principales: Acceso a Datos, Lógica del Negocio, y Presentación. El subsistema Acceso a Datos contiene las clases encargadas de la persistencia y recuperación de los ejercicios de ordenamiento de tarjetas. El subsistema Lógica del Negocio está compuesto a su vez por los subsistemas: ES (Entrada y Salida), Procesos, y Entidades. El subsistema Procesos está compuesto por los subsistemas: Preparación, Ejecución y Evaluación. El subsistema Presentación, a su vez, está compuesto por los subsistemas Preparación EOT, Ejecución EOT y Evaluación EOT. A continuación se muestra una vista global del modelo de implementación de la solución propuesta (Fig. 23).

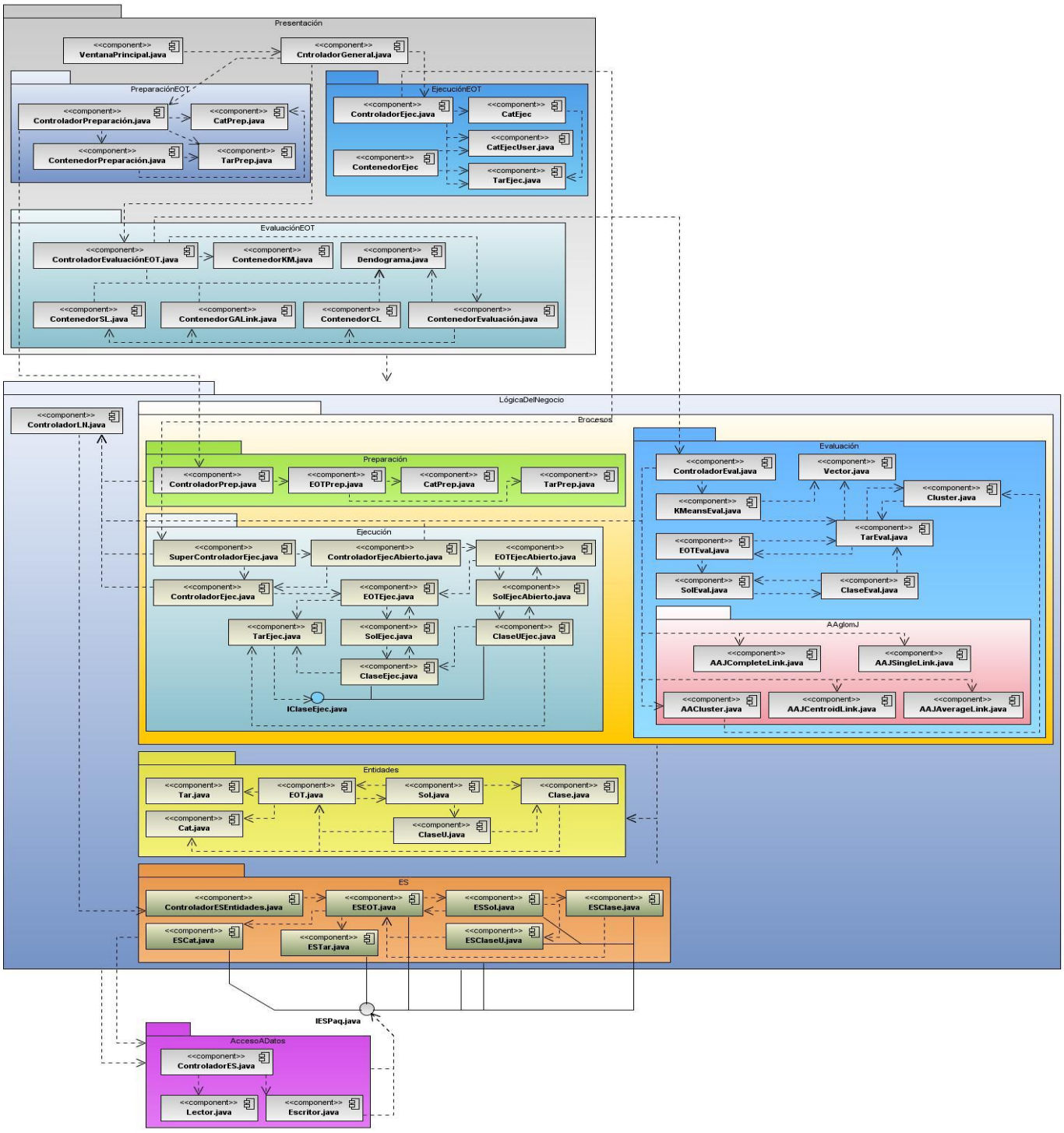


Fig. 23. Diagrama de Componentes General

Prueba

Caso de Uso del Sistema Preparar Ejercicio de Ordenamiento de Tarjetas

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Crear EOT.	EC 1.1: Insertar el nombre y la descripción del EOT correctamente.	Esta funcionalidad crea un EOT.	1. Hacer clic sobre el botón nuevo EOT, o en el menú Archivo-Nuevo Ejercicio. 2. Introducir el nombre del EOT en el campo homónimamente etiquetado. 3. Seleccionar el tipo de EOT. 4. Introducir las orientaciones o descripción del EOT en el campo homónimamente etiquetado. 5. Hacer clic en el botón Aceptar. 6. Se crea el EOT.
	EC 1.2: Insertar el nombre, dejando la descripción en blanco.		1. Hacer clic sobre el botón nuevo Ejercicio, o en el menú Archivo-Nuevo Ejercicio. 2. Introducir el nombre del EOT en el campo homónimamente etiquetado. 3. Seleccionar el tipo de EOT. 4. Hacer clic en el botón Aceptar. 5. Se crea el EOT.
	EC 1.3: Insertar la descripción, dejando el nombre en blanco.		1. Hacer clic sobre el botón nuevo EOT, o en el menú Archivo-Nuevo EOT. 2. Seleccionar el tipo de EOT. 3. Introducir la descripción del EOT en el campo homónimamente etiquetado. 4. Hacer clic en el botón Aceptar. 5. Se muestra un mensaje de error.

V1: Nombre, V2: Descripción (V-válido, I-inválido, N/A-no es necesario)

Id del escenario	Escenario	V1	V2	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Insertar el nombre y la descripción del EOT correctamente.	V	V	El sistema genera un nuevo EOT y muestra la interfaz de preparación de EOT.	Satisfactorio
EC 1.2	Insertar el nombre, dejando la descripción en blanco.	V	N/A	El sistema genera un nuevo EOT y muestra la interfaz de preparación de EOT.	Satisfactorio
EC 1.3	Insertar la descripción, dejando el nombre en blanco.	I	V	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informando al usuario que el nombre es inválido.	Satisfactorio

Caso de Uso del Sistema Gestionar tarjetas

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Crear Tarjeta.	EC 1.1: Insertar el nombre y la descripción correctamente.	Esta funcionalidad crea una tarjeta.	1. Hacer clic sobre el botón "Nueva Tarjeta". 2. Introducir el nombre de la Tarjeta en el campo homónimamente etiquetado. 3. Seleccionar el checkbox etiquetado "Descripción". 4. Introducir la descripción de la tarjeta en el campo homónimamente etiquetado. 5. Hacer clic en el botón "Agregar". 6. El sistema muestra la nueva tarjeta.
	EC 1.2: Insertar el nombre, dejando la descripción en blanco.		1. Hacer clic sobre el botón "Nueva Tarjeta". 2. Introducir el nombre de la tarjeta en el campo homónimamente etiquetado. 3. Hacer clic en el botón "Agregar". 4. El sistema muestra la nueva tarjeta.
	EC 1.3: Insertar la descripción, dejando el nombre en blanco.		1. Hacer clic sobre el botón "Nueva Tarjeta". 2. Seleccionar el checkbox etiquetado "Descripción". 3. Introducir la descripción de la tarjeta en el campo homónimamente etiquetado. 4. Hacer clic en el botón "Agregar". 5. El sistema muestra un mensaje de error.
Modificar Tarjeta.	EC 2.1: Modificar el nombre y la descripción correctamente.	Esta funcionalidad modifica el contenido de una Tarjeta.	1. Hacer clic sobre la tarjeta deseada. 2. Seleccionar "Editar información de la Tarjeta". 3. Reemplazar el valor del campo etiquetado "Nombre". 4. Seleccionar o desmarcar el checkbox etiquetado "Descripción". 5. Reemplazar el valor del campo etiquetado "Descripción". 6. Hacer clic en el botón "Aceptar". 7. El sistema muestra la tarjeta con el contenido modificado.
	EC 2.2: Modificar el nombre, dejando la descripción en blanco.		1. Hacer clic sobre la tarjeta deseada. 2. Seleccionar "Editar Tarjeta". 3. Reemplazar el valor del campo etiquetado "Nombre". 4. Seleccionar o desmarcar el checkbox etiquetado "Descripción". 5. Reemplazar el valor del campo etiquetado "Descripción". 6. Hacer clic en el botón "Aceptar". 7. El sistema muestra la tarjeta con el contenido modificado.
	EC 2.3: Modificar la descripción, dejando el nombre en blanco.		1. Hacer clic sobre la tarjeta deseada. 2. Seleccionar "Editar Tarjeta". 3. Reemplazar el valor del campo etiquetado "Nombre". 4. Seleccionar o desmarcar el checkbox etiquetado "Descripción". 5. Reemplazar el valor del campo etiquetado "Descripción". 6. Hacer clic en el botón "Aceptar". 7. El sistema muestra un mensaje de error.
Eliminar Tarjeta.	EC 3.1: Eliminar Tarjeta.	Esta funcionalidad elimina una Tarjeta.	1. Hacer clic sobre la tarjeta deseada. 2. Hacer clic sobre el botón "Eliminar Tarjeta". 3. El sistema solicita confirmar la acción 4. Confirmar la acción. 5. El sistema elimina la tarjeta.

V1: Nombre, V2: Descripción (V-válido, I-inválido, N/A-no es necesario)

Id del escenario	Escenario	V1	V2	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Insertar el nombre y la descripción del EOT correctamente.	V	V	El sistema genera una nueva tarjeta y la muestra.	Satisfactorio
EC 1.2	Insertar el nombre, dejando la descripción en blanco.	V	N/A	El sistema genera una nueva tarjeta y la muestra.	Satisfactorio
EC 1.3	Insertar la descripción, dejando el nombre en blanco.	I	V	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informando al usuario que el nombre es inválido.	Satisfactorio
EC 2.1	Modificar el nombre, dejando la descripción en blanco.	V	N/A	El sistema modifica el nombre de la tarjeta y la muestra.	Satisfactorio
EC 2.2	Modificar la descripción, dejando el nombre en blanco.	V	N/A	El sistema modifica el nombre de la tarjeta y la muestra.	Satisfactorio
EC 2.3	Modificar la descripción, dejando el nombre en blanco.	I	V	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informando al usuario que el nombre es inválido.	Satisfactorio
EC 3.1	Eliminar Tarjeta	N/A	N/A	El sistema solicita confirmación y elimina la tarjeta.	Satisfactorio

Caso de Uso del Sistema Gestionar categorías

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Crear categoría.	EC 1.1: Insertar el nombre y la descripción correctamente.	Esta funcionalidad crea una categoría.	1. Hacer clic sobre el botón nueva categoría. 2. Introducir el nombre de la categoría en el campo homónimamente etiquetado. 3. Seleccionar el checkbox etiquetado "Descripción". 4. Introducir la descripción de la categoría en el campo homónimamente etiquetado. 5. Hacer clic en el botón "Agregar". 6. El sistema muestra la nueva categoría.
	EC 1.2: Insertar el nombre, dejando la descripción en blanco.		1. Hacer clic sobre el botón nueva categoría. 2. Introducir el nombre de la categoría en el campo homónimamente etiquetado. 3. Hacer clic en el botón "Agregar". 4. El sistema muestra la nueva categoría.
	EC 1.3: Insertar la descripción, dejando el nombre en blanco.		1. Hacer clic sobre el botón nueva categoría. 2. Seleccionar el checkbox etiquetado "Descripción". 3. Introducir la descripción de la categoría en el campo homónimamente etiquetado. 4. Hacer clic en el botón "Agregar". 5. El sistema muestra un mensaje de error.

Modificar categoría.	EC 2.1: Modificar el nombre y la descripción correctamente.	Esta funcionalidad modifica el contenido de una categoría.	1. Hacer clic sobre la categoría deseada. 2. Seleccionar "Editar categoría". 3. Reemplazar el valor del campo etiquetado "Nombre". 4. Seleccionar o desmarcar el checkbox etiquetado "Descripción". 5. Reemplazar el valor del campo etiquetado "Descripción". 6. Hacer clic en el botón "Aceptar". 7. El sistema muestra la categoría con el contenido modificado.
	EC 2.2: Modificar el nombre, dejando la descripción en blanco.		1. Hacer clic sobre la categoría deseada. 2. Seleccionar "Editar categoría". 3. Reemplazar el valor del campo etiquetado "Nombre". 4. Seleccionar o desmarcar el checkbox etiquetado "Descripción". 5. Reemplazar el valor del campo etiquetado "Descripción". 6. Hacer clic en el botón "Aceptar". 7. El sistema muestra la categoría con el contenido modificado.
	EC 2.3: Modificar la descripción, dejando el nombre en blanco.		1. Hacer clic sobre la categoría deseada. 2. Seleccionar "Editar categoría". 3. Reemplazar el valor del campo etiquetado "Nombre". 4. Seleccionar o desmarcar el checkbox etiquetado "Descripción". 5. Reemplazar el valor del campo etiquetado "Descripción". 6. Hacer clic en el botón "Aceptar". 7. El sistema muestra un mensaje de error.
Eliminar categoría.	EC 3.1: Eliminar categoría.	Esta funcionalidad elimina una categoría.	1. Hacer clic sobre la categoría deseada. 2. Hacer clic sobre el botón "Eliminar categoría". 3. El sistema solicita confirmar la acción 4. Confirmar la acción. 5. El sistema elimina la categoría.

V1: Nombre, V2: Descripción (V-válido, I-inválido, N/A-no es necesario)

Id del escenario	Escenario	V1	V2	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Insertar el nombre y la descripción del EOT correctamente.	V	V	El sistema genera una nueva categoría y la muestra.	Satisfactorio
EC 1.2	Insertar el nombre, dejando la descripción en blanco.	V	N/A	El sistema genera una nueva categoría y la muestra.	Satisfactorio
EC 1.3	Insertar la descripción, dejando el nombre en blanco.	I	V	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informando al usuario que el nombre es inválido.	Satisfactorio
EC 2.1	Modificar el nombre, dejando la descripción en blanco.	V	N/A	El sistema modifica el nombre de la categoría y la muestra.	Satisfactorio
EC 1.3	Modificar la descripción, dejando el nombre en blanco.	V	N/A	El sistema modifica el nombre de la categoría y la muestra.	Satisfactorio

EC 2.1	Modificar la descripción, dejando el nombre en blanco.	I	V	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informando al usuario que el nombre es inválido.	Satisfactorio
EC 3.1	Eliminar categoría	N/A	N/A	El sistema solicita confirmación y elimina la categoría.	Satisfactorio

Caso de Uso del Sistema Ejecutar Ejercicio de ordenamiento de tarjetas

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Ejecutar EOT.	EC 1.1: Asignar a una clase una tarjeta procedente del montón.	Esta funcionalidad agrupa las tarjetas en clases.	1. Arrastrar la tarjeta desde el montón hacia la clase deseada.
	EC 1.2: Asignar a una clase una tarjeta procedente de otra clase.		1. Arrastrar la tarjeta seleccionada desde la clase origen hacia la clase destino.
	EC 1.3: Desagrupar tarjeta.	Esta funcionalidad desagrupa la tarjeta, es decir, la mueve de la clase hacia el montón un EOT.	1. Arrastrar la tarjeta seleccionada desde la clase hacia el montón.

V1: Tarjeta, V2: clase (V-válido, I-inválido, N/A-no es necesario)

Id del escenario	Escenario	V1	V2	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Asignar a una clase una tarjeta procedente del montón.	N/A	V	El sistema muestra la tarjeta dentro de la clase.	Satisfactorio.
EC 1.2	Asignar a una clase una tarjeta procedente de otra clase.	N/A	V	El sistema elimina la tarjeta en su clase original la muestra dentro de la clase destino.	Satisfactorio.
EC 1.3	Desagrupar tarjeta.	V	N/A	El sistema elimina la tarjeta en su clase original la muestra dentro del montón.	Satisfactorio.

Caso de Uso del Sistema Evaluar ejercicio de ordenamiento de tarjetas Resueltos

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Mezclar EOT.	EC 1.1: Seleccionar ficheros con soluciones correspondientes a un mismo EOT.	Esta funcionalidad mezcla las soluciones incluidas en diferentes ficheros de un mismo EOT.	1. Seleccionar "Mezclar EOT". 2. Seleccionar los ficheros a Mezclar. 3. El sistema modifica la composición de los ficheros de EOT. 4. El sistema muestra un mensaje informando si la acción tuvo o no éxito. 5. En caso de haberse mezclado los ejercicios correctamente el sistema muestra un mensaje informando si se desea actualizar las evaluaciones realizadas hasta el momento con las nuevas soluciones.
	EC 1.2: Seleccionar ficheros con soluciones correspondientes a diferentes EOT.		1. Seleccionar "Mezclar EOT". 2. Seleccionar los ficheros a Mezclar. 3. El sistema muestra un mensaje de error.
Evaluar EOT.	EC 2.1: Marcar el checkbox etiquetado "Single - Link".	Esta funcionalidad evalúa las soluciones de un EOT mediante el algoritmo Single - Link.	1. Seleccionar el checkbox etiquetado "Single – Link". 2. El sistema muestra un dendograma con los resultados del algoritmo.
	EC 2.2: Marcar el checkbox etiquetado "Complete - Link".	Esta funcionalidad evalúa las soluciones de un EOT mediante el algoritmo Single - Link.	1. Seleccionar el checkbox etiquetado "Complete – Link". 2. El sistema muestra un dendograma con los resultados del algoritmo.
	EC 2.3: Marcar el checkbox etiquetado "Group Average - Link".	Esta funcionalidad evalúa las soluciones de un EOT mediante el algoritmo Group Average - Link.	1. Seleccionar el checkbox etiquetado "Group Average – Link". 2. El sistema muestra un dendograma con los resultados del algoritmo.
	EC 2.4: Marcar el checkbox etiquetado "K - Means", e introducir una cantidad de clústeres menor que 1, y una cantidad de iteraciones mayor que 1, dejando los otros valores por defecto.	Esta funcionalidad evalúa las soluciones de un EOT mediante el algoritmo K - Means.	1. Seleccionar el checkbox etiquetado "K – Means". 2. Introducir una cantidad de clústeres menor que cero. 3. Introducir una cantidad de iteraciones mayor que cero. 4. El sistema muestra un mensaje de error.
	EC 2.5: Marcar el checkbox etiquetado "K - Means", e introducir una cantidad de clústeres mayor que 1, y una cantidad de iteraciones menor que 1, dejando los otros valores por defecto.		1. Seleccionar el checkbox etiquetado "K – Means". 2. Introducir una cantidad de clústeres mayor que cero. 3. Introducir una cantidad de iteraciones menor que cero. 4. El sistema muestra un mensaje de error.
	EC 2.6: Marcar el checkbox etiquetado "K - Means", e introducir una cantidad de clústeres mayor que 1, y una cantidad de iteraciones mayor que 1, dejando los otros valores por defecto.		1. Seleccionar el checkbox etiquetado "K – Means". 2. Introducir una cantidad de clústeres mayor que cero. 3. Introducir una cantidad de iteraciones mayor que cero. 4. Hacer clic en sobre el botón "Aceptar". 5. El sistema muestra los agrupamientos resultantes del algoritmo.

	EC 2.7: Marcar el checkbox etiquetado "K - Means", e introducir valores mayores que 1 en los campos "Cantidad de iteraciones" y "Cantidad de Clústeres". Introducir valores menores que cero en el campo "SRC", dejando el campo "Variación de SRC" con su valor por defecto.		1. Seleccionar el checkbox etiquetado "K – Means". 2. Introducir una cantidad de clústeres mayor que cero. 3. Introducir una cantidad de iteraciones mayor que cero. 4. Introducir un valor SRC menor que cero. 5. Hacer clic en sobre el botón "Aceptar". 6. El sistema muestra un mensaje de error.
	EC 2.8: Marcar el checkbox etiquetado "K - Means", e introducir valores mayores que cero en los campos "Cantidad de iteraciones" y "Cantidad de Clústeres". Introducir valores menores que cero en el campo "Variación de SRC", dejando el campo "SRC" con su valor por defecto.		1. Seleccionar el checkbox etiquetado "K – Means". 2. Introducir una cantidad de clústeres mayor que cero. 3. Introducir una cantidad de iteraciones mayor que cero. 4. Introducir un valor SRC menor que cero. 5. Hacer clic en sobre el botón "Aceptar". 6. El sistema muestra un mensaje de error.
	EC 2.9: Marcar el checkbox etiquetado "K - Means", e introducir valores mayores que cero en los campos "Cantidad de iteraciones", "Cantidad de Clústeres" y "SRC", y un valor mayor que cero en el campo "Variación de SRC".		1. Seleccionar el checkbox etiquetado "K – Means". 2. Introducir una cantidad de clústeres mayor que cero. 3. Introducir una cantidad de iteraciones mayor que cero. 4. Introducir un valor SRC mayor que cero. 5. Introducir un valor de variación de SRC mayor que cero. 6. Hacer clic en sobre el botón "Aceptar". 7. El sistema muestra los agrupamientos resultantes del algoritmo.

V1: Cantidad de iteraciones, V2: Cantidad de Clústeres, V3: SRC, V4: Variación de SRC (V-válido, I-inválido, N/A-no es necesario)

Id del escenario	Escenario	V1	V2	V3	V4	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Seleccionar ficheros con soluciones correspondientes a un mismo EOT.	N/A	N/A	N/A	N/A	El sistema varía la composición de las soluciones en los ficheros y muestra un mensaje informando el éxito de la operación.	Satisfactorio.
EC 1.2	Seleccionar ficheros con soluciones correspondientes a diferentes EOT.	N/A	N/A	N/A	N/A	El sistema muestra un mensaje de error.	Satisfactorio.
EC 2.1	Marcar el checkbox etiquetado "Single - Link".	N/A	V	N/A	V	El sistema muestra un dendograma con los resultados del algoritmo.	Satisfactorio.
EC 2.2	Marcar el checkbox etiquetado "Complete - Link".	N/A	V	N/A	V	El sistema muestra un dendograma con los resultados del algoritmo.	Satisfactorio.
EC 2.3	Marcar el checkbox etiquetado "Average - Link".	N/A	V	N/A	V	El sistema muestra un dendograma con los resultados del algoritmo.	Satisfactorio.

EC 2.4	Marcar el checkbox etiquetado “K - Means”, e introducir una cantidad de clústeres menor que 1, y una cantidad de iteraciones mayor que 1, dejando los otros valores por defecto.	I	V	V	V	El sistema verifica el valor insertado y muestra un mensaje de error.	Satisfactorio.
EC 2.5	Marcar el checkbox etiquetado “K - Means”, e introducir una cantidad de clústeres mayor que 1, y una cantidad de iteraciones menor que 1, dejando los otros valores por defecto.	V	I	V	V	El sistema verifica el valor insertado y muestra un mensaje de error.	Satisfactorio.
EC 2.6	Marcar el checkbox etiquetado “K - Means”, e introducir una cantidad de clústeres mayor que 1, y una cantidad de iteraciones mayor que 1, dejando los otros valores por defecto.	V	V	V	V	El sistema muestra los agrupamientos obtenidos por el algoritmo.	Satisfactorio.
EC 2.7	Marcar el checkbox etiquetado “K - Means”, e introducir valores mayores que 1 en los campos “Cantidad de iteraciones” y “Cantidad de Clústeres”. Introducir valores menores que cero en el campo “SRC”, dejando el campo “Variación de SRC” con su valor por defecto.	V	V	I	V	El sistema verifica el valor insertado y muestra un mensaje de error.	Satisfactorio.
EC 2.8	Marcar el checkbox etiquetado “K - Means”, e introducir valores mayores que cero en los campos “Cantidad de iteraciones” y “Cantidad de Clústeres”. Introducir valores menores que cero en el campo “Variación de SRC”, dejando el campo “SRC” con su valor por defecto.	V	V	V	I	El sistema verifica el valor insertado y muestra un mensaje de error.	Satisfactorio.
EC 2.9	Marcar el checkbox etiquetado “K - Means”, e introducir valores mayores que cero en los campos “Cantidad de iteraciones”, “Cantidad de Clústeres” y “SRC”, y un valor mayor que cero en el campo “Variación de SRC”.	V	V	V	V	El sistema muestra los agrupamientos obtenidos por el algoritmo.	Satisfactorio.

Caso de Uso del Sistema Almacenar Ejercicio de ordenamiento de tarjetas

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Almacenar EOT.	EC 1.1: Seleccionar una ubicación y el formato ".eot" para crear un fichero nuevo.	Esta funcionalidad garantiza la persistencia de un EOT.	1. Seleccionar Guardar EOT 2. Seleccionar formato ".eot" 3. Seleccionar ubicación e introducir el nombre del fichero. 4. Hacer clic sobre el botón "Guardar". 5. El sistema almacena el EOT en el fichero.
	EC 1.2: Seleccionar una ubicación y el formato ".eot" para reescribir un fichero existente.		1. Seleccionar Guardar como EOT 2. Seleccionar formato ".eot" 3. Seleccionar el fichero. 4. Hacer clic sobre el botón "Guardar". 5. El sistema solicita la confirmación. 6. El sistema sobrescribe el fichero.
	EC 1.3: Seleccionar una ubicación y el formato ".xml" para crear un fichero nuevo.	Esta funcionalidad garantiza la persistencia de un EOT.	1. Seleccionar Guardar EOT 2. Seleccionar formato ".xml" 3. Seleccionar ubicación e introducir el nombre del fichero. 4. Hacer clic sobre el botón "Guardar". 5. El sistema almacena el EOT en el fichero.
	EC 1.4: Seleccionar una ubicación y el formato ".xml" para reescribir un fichero existente.		1. Seleccionar Guardar como EOT 2. Seleccionar formato ".xml" 3. Seleccionar el fichero. 4. Hacer clic sobre el botón "Guardar". 5. El sistema solicita la confirmación. 6. El sistema sobrescribe el fichero.

V1: Formato del fichero, V2: Ubicación del fichero, V3: Nombre del fichero (V-válido, I-inválido, N/A-no es necesario)

Id del escenario	Escenario	V1	V2	V3	V4	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Seleccionar una ubicación y el formato ".eot" para crear un fichero nuevo.	V	V	N/A	V	El sistema genera un nuevo fichero con el nombre y formato indicados en la ubicación señalada	Satisfactorio.
EC 1.2	Seleccionar una ubicación y el formato ".eot" para reescribir un fichero existente.	V	V	N/A	V	El sistema reescribe el fichero con el nombre y formato indicados en la ubicación señalada.	Satisfactorio.
EC 1.3	Seleccionar una ubicación y el formato ".xml" para crear un fichero nuevo.	V	V	N/A	V	El sistema genera un nuevo fichero con el nombre y formato indicados en la ubicación señalada	Satisfactorio.
EC 1.4	Seleccionar una ubicación y el formato ".xml" para reescribir un fichero existente.	V	V	N/A	V	El sistema reescribe el fichero con el nombre y formato indicados en la ubicación señalada.	Satisfactorio.

Caso de Uso del Sistema Recuperar Ejercicio de ordenamiento de tarjetas

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Recuperar EOT.	EC 1.1: Seleccionar un fichero con extensión y formato ".eot".	Esta funcionalidad garantiza la recuperación de un EOT.	1. Seleccionar Abrir EOT 2. Seleccionar fichero. Con formato ".eot" 3. Hacer clic sobre el botón "Abrir". 4. El sistema muestra el EOT contenido en el fichero.
	EC 1.2: Seleccionar un fichero con extensión ".xml" y formato ".eot",		1. Seleccionar Abrir EOT 2. Seleccionar fichero. Con formato ".eot" 3. Hacer clic sobre el botón "Abrir". 4. El sistema muestra el EOT contenido en el fichero.
	EC 1.3: Seleccionar un fichero con extensión ".eot" y formato ".xml".		1. Seleccionar Abrir EOT 2. Seleccionar fichero. Con formato ".eot" 3. Hacer clic sobre el botón "Abrir". 4. El sistema muestra el EOT contenido en el fichero.
	EC 1.4: Seleccionar un fichero con extensión no válida y formato no válido.		1. Seleccionar Abrir EOT 2. Seleccionar fichero. Con formato ".eot" o ".xml" 3. Hacer clic sobre el botón "Abrir". 4. El sistema muestra un mensaje de error.
	EC 1.5: Seleccionar un fichero con extensión y formato ".xml".		1. Seleccionar Abrir EOT 2. Seleccionar fichero. Con formato ".xml" 3. Hacer clic sobre el botón "Abrir". 4. El sistema muestra el EOT contenido en el fichero.

V1: Formato del fichero, V2: Ubicación del fichero, V3: Extensión del fichero (V-válido, I-inválido, N/A-no es necesario)

Id del escenario	Escenario	V1	V2	V3	V4	Respuesta del Sistema	Resultado de la Prueba
EC 1.1	Seleccionar un fichero con extensión y formato ".eot".	V	V	N/A	V	El sistema muestra el EOT contenido en el fichero.	Satisfactorio.
EC 1.2	Seleccionar un fichero con extensión .xml y formato ".eot".	V	V	N/A	V	El sistema muestra el EOT contenido en el fichero.	Satisfactorio.
EC 1.3	Seleccionar un fichero con extensión .eot y formato ".xml".	V	V	N/A	V	El sistema muestra el EOT contenido en el fichero.	Satisfactorio.
EC 1.4	Seleccionar un fichero con extensión no válida y formato no válido.	I	I	N/A	V	El sistema muestra un mensaje de error.	Satisfactorio.
EC 1.5	Seleccionar un fichero con extensión y formato ".xml".	V	V	N/A	V	El sistema muestra el EOT contenido en el fichero.	Satisfactorio.

Conclusiones del capítulo

Siguiendo la arquitectura de tres capas asumida para el sistema se definieron tres subsistemas. El subsistema Acceso a Datos, que puede ser reutilizado en el desarrollo de otras herramientas de la plataforma ABAD. El subsistema Lógica del Negocio, que puede ser reutilizado para escalar la solución hacia la Web, pues es independiente del subsistema Presentación. Este último subsistema puede ser escalado para mostrar nuevas formas de representar los resultados de la técnica. Asimismo, la definición de un fichero con formato de XML contribuye a la compatibilidad con el resto de la plataforma ABAD.

Conclusiones

El sistema propuesto en el presente trabajo de diploma automatiza la técnica de ordenamiento de tarjetas, la cual constituye una herramienta útil para incrementar la efectividad y la eficiencia de la arquitectura de la información, mediante la estructuración de contenidos basados en los Métodos de Diseño Centrados en el Usuario.

El desarrollo de la presente solución informática para la automatización del ordenamiento de tarjetas permitió dar cumplimiento a los objetivos trazados al inicio de esta investigación, con la cual se pudo arribar a las siguientes conclusiones:

- La aplicación de la técnica de ordenamiento de tarjetas genera grandes cantidades de datos, difíciles de procesar estadísticamente de forma manual.
- El sistema elaborado abarca las fases de preparación, ejecución y evaluación del ordenamiento de tarjetas, distinguiéndose por la implementación del algoritmo de formación de aglomerados planos K – Means.
- La arquitectura del sistema permite la escalabilidad y la reutilización de sus componentes por otras herramientas de la plataforma ABAD.
- La persistencia de las entidades del negocio a través de un fichero en formato XML contribuye a la compatibilidad con otras herramientas de la plataforma ABAD.
- La solución elaborada contribuye a reducir los errores y demoras en la fase de evaluación del ordenamiento de tarjetas, a la vez que establece un estándar de evaluación que se ajusta a las necesidades de diseño de la arquitectura de información para el desarrollo de soluciones informáticas en la UCI.

Recomendaciones

Esta solución informática es el primer paso de la plataforma para la arquitectura de la información ABAD, por lo que se recomienda:

- Extender las funcionalidades del módulo de Evaluación de EOT, mediante la implementación de la captura manual de datos de ejercicios de ordenamiento de tarjetas ejecutados sin hacer uso del módulo de Ejecución de EOT.
- Extender las funcionalidades del módulo de Evaluación de EOT, mediante la compatibilización con los formatos utilizados por herramientas foráneas para la realización de la técnica de ordenamiento de tarjetas y sistemas de hojas de cálculo.
- Extender las funcionalidades del módulo de Evaluación de EOT, mediante la implementación del análisis de secuencia de los ordenamientos.
- Extender las funcionalidades del módulo de Evaluación de EOT, mediante la implementación del análisis de frecuencia de denominaciones de clases, de tal forma que la herramienta brinde recomendaciones para las denominaciones de los agrupamientos.
- Extender las funcionalidades del módulo de Evaluación de EOT, mediante técnicas de representación compatibles con la inversión provocada por el algoritmo Centroid - Link.

Bibliografía

Ailonwebs. 2009. Aplicaciones Web. [En línea] 2009. http://www.ailonwebs.com/aplicaciones-web.php.

Eclipse. 2009. SWT: The Standard Widget Toolkit. [En línea] Eclipse, 2009. http://www.eclipse.org/swt/.

Española, Real Academia. Diccionario de la lengua española - Vigésima segunda edición. *Real Academia Española.* [En línea] Real Academia Española. [Citado: diciembre 11, 2009.] http://www.rae.es/rae-index.html.

Flower, Martin. 2003. *Patterns of Enterprise Application Architecture.* s.l. : Addison Wesley, 2003.

Garrett, Jesse James. 2002. *The Elements of User Experience.* New York : New Riders Publishing, 2002.

Grupo de Ingeniería del Software y Sistemas de Información. 2003. Ingeniería del Software y Sistemas de Información. [En línea] 2003. http://issi.dsic.upv.es/publications/archives/f-1069167248521/actas.pdf.

IBM. 2010. Rational Rose Enterprise. *IBM.* [En línea] 2 12, 2010. [Citado: 2 12, 2010.] http://www-01.ibm.com/software/awdtools/developer/rose/enterprise/features/index.html?S_CMP=rnav.

Instituto Tecnológico de Hermosillo. 2009. El lenguaje Java. [En línea] 2009. http://eddi.ith.mx/Curso/Contenido/java.htm.

Jacobson, Ivar, Boch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley, 2000.

JAVA. 2009. JAVA. [En línea] 2009. http://java.com/es/about/.

Beck, Kent y Andres, Cyntia. 1999. *Extreme Programming Explained.* s.l. : Addison-Wesley Professional, 1999.

Beck, Kent, et al. 2001. Manifesto for Agile Software Development. [En línea] 2001. http://agilemanifesto.org/.

Larman, Craig. 1999. *UML y Patrones.* s.l. : Patience Hall, 1999. 970-17-0261-1.

Lime & Chile. 2009. WebSort. [En línea] 2009. http://websort.net/.

Manning, Christopher D. 2007. *An Introduction to Information Retrieval.* Cambridge : Cambridge University Press, 2007.

Marinilli, Mauro. 2006. Swing and SWT: A Tale of Two Java GUI Libraries. [En línea] 2006. http://www.developer.com/java/other/artide.php/2179061/Swing-and-SWT-A-Tale-of-Two-Java-GUI-Libraries.htm.

Mercer, Dave. 2001. Fundamentos de Programación en XML. *Blioteca.uci.cu.* [En línea] 10 2001. [Citado: 3 26, 2010.] http://bibliodoc.uci.cu/pdf/reg01311.pdf. 958-41- 0297- 4.

Montes de Oca, Antonio. 2004. *Arquitectura de información y usabilidad: nociones básicas para los.* [EBSCO] La Habana : Acimed, 2004.

OpenXava. 2009. OpenXava. [En línea] 2009. http://www.gestion400.com/web/.

OPTIC . 2009. Normas y Estándares de Portales Gubernamentales República Dominicana. [En línea] 2009. http://www.sap.gob.do/LinkClick.aspx?fileticket=NuzJ%2Bqs48Hg%3D&tabid=63&mid=420&language=es-ES.

OptimalSort. 2009. OptimalSort. [En línea] 2009. http://www.optimalsort.com/.

Orovio Pino, Jeison Mario. 2009. *Análisis y Diseño de un sistema para la aplicación de técnicas de Card Sorting en la obtención de Arquitecturas de información.* La Habana : Universidad de las Ciencias Informáticas, 2009.

Paul, Celeste Lyn. 2008. *A Modified Delphi Approach to a New Card Sorting Methodology.* [Journal of Usability Studies] Ashburn, Virginia : Usability Professionals' Association, 2008.

Peter Morville, Louis Rosenfeld. 2006. *Information Architecture for the World Wide Web, Third Edition.* s.l. : O'Reilly Media, 2006.

Python. 2009. Python.org. [En línea] 2009. http://www.python.org/about/.

Reynoso, Billy. Architect Academy: Seminario de Arquitectura de Software. [En línea] http://download.microsoft.com/download/5/6/8/568d7ce9-d0ca-46d6-a6cb-8b53cf10b134/050608-Architect%20Academy%20Webcast%201.ppt.

Sola Padilla, Yannelys. 2008. *Análisis de un sistema automatizado a través de la técnica de Card Sorting u Ordenación de Tarjetas.* La Habana : Universidad de las Ciencias Informáticas, 2008.

Soulie, J. 2009. Cplusplus. [En línea] 2009. http://www.cplusplus.com/doc/tutorial/.

Spencer, Donna. 2009. *Card Sorting Designing Usable Categories.* New York : Rosenfeld Media, 2009.

Sun Microsystems. 2010. Sun Developer Network. [En línea] Mayo 2010. http://java.sun.com/javase/.

Syntagm. 1995. Syntagm. [En línea] 1995. http://www.syntagm.co.uk/design/cardsortintro.shtml.

TEO, JAVIER MOLDES. 2003. *Java 2 J2se 1.4 Guias Practicas .* s.l. : Anaya Multimedia, 2003.

UCI. 2010. Entorno Virtual de Aprendizaje. *Práctica Profesional.* [En línea] UCI, 3 25, 2010. [Citado: 3 25, 2010.] http://eva.uci.cu/mod/resource/view.php?id=29419.

Under, Wille. 2009. [En línea] mayo 2009. http://faler.wordpress.com/2009/05/29/why-i-switched-from-eclipse-to-netbeans/.

UsabilityCentric. 2009. UXSort.com. [En línea] 2009. http://uxsort.com/.

Visual Paradigm. 2010. Visual Paradigm for UML 7.4. *Visual Paradigm.* [En línea] 2 12, 2010. [Citado: 2 12, 2010.] http://images.visual-paradigm.com/datasheets/vpuml72_datasheet.pdf.

Warfel, Todd. 2004. Card sorting: a definitive guide. *Box and Arrows.* [En línea] April 7, 2004. [Citado: December 2, 2009.] http://www.boxesandarrows.com/view/card_sorting_a_definitive_guide.

Wells, Don. 2009. Extreme Programming: A gentle introduction. [En línea] 2009. http://www.extremeprogramming.org/rules.html.

Wurman, Richard Saul. 1997. *Information Architecture.* Los Ángeles : Watson-Guptill Pubis, 1997.

Zukowski, John. 2003. *Java 2 J2SE 1.4.* s.l. : Anaya Multimedia, 2003.

Anexos

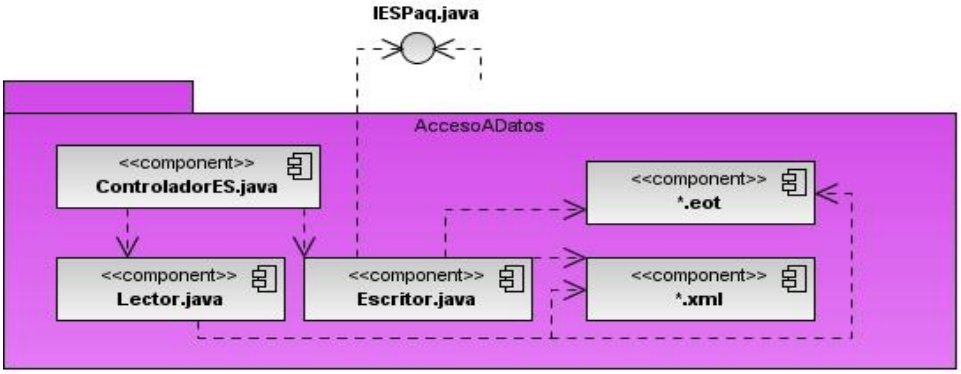


Fig. 24. Subsistema Acceso a Datos

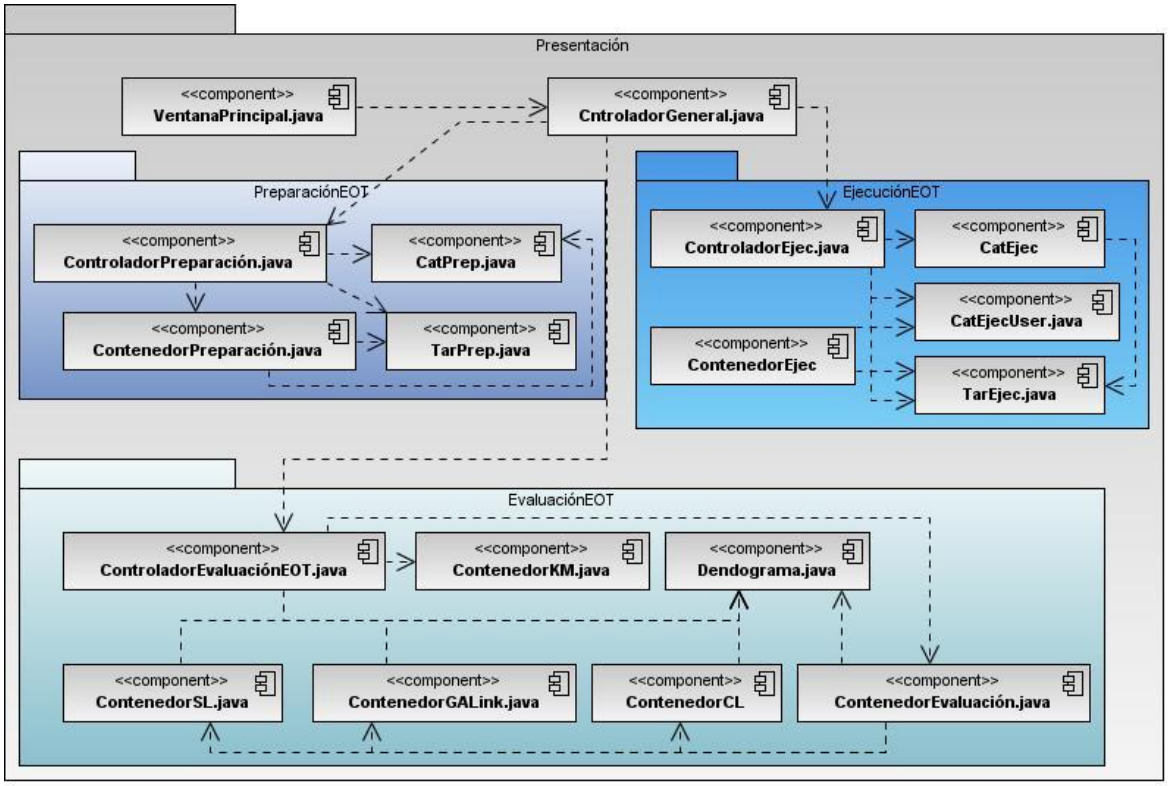


Fig. 29. Subsistema Presentación

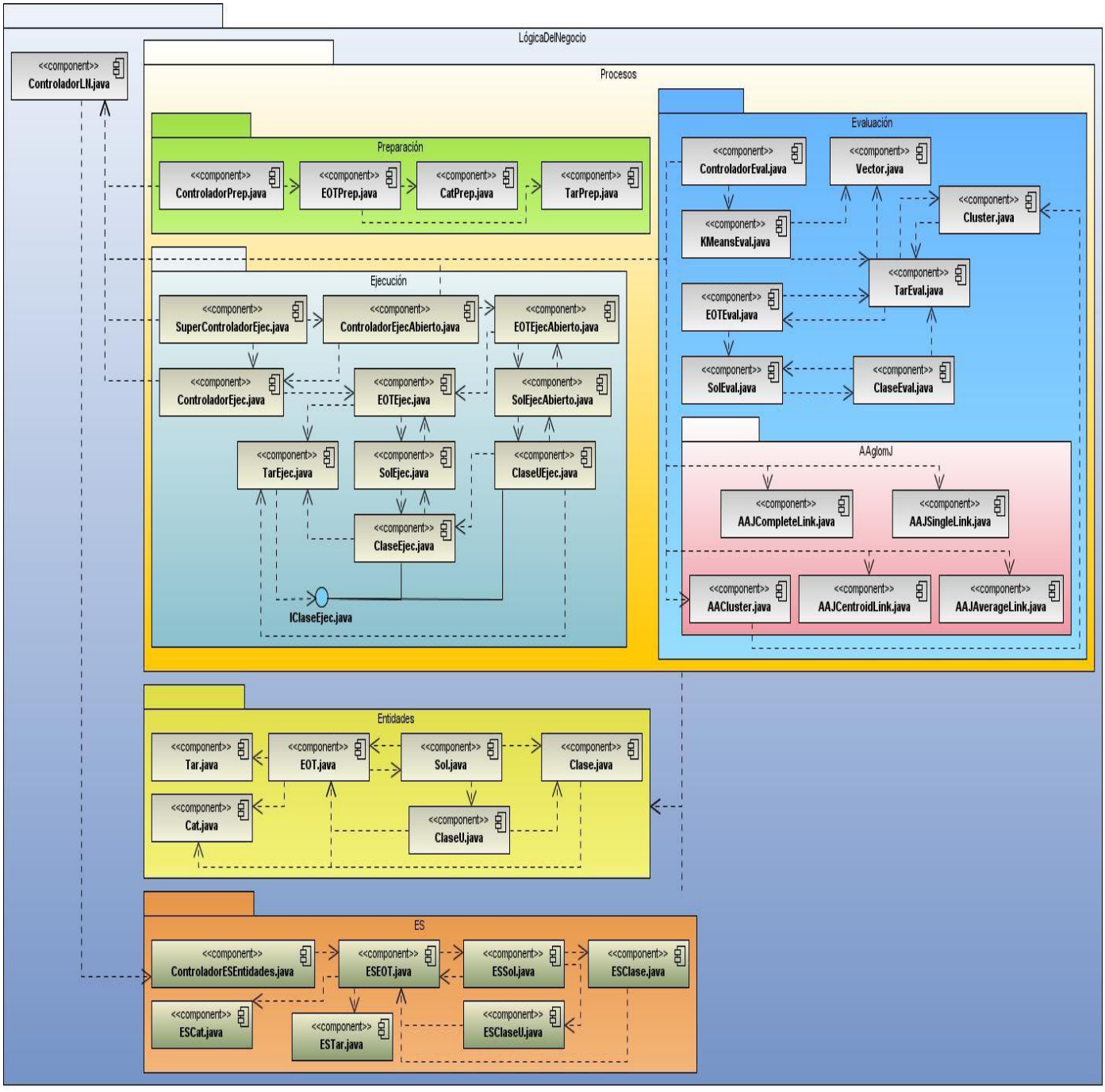


Fig. 30. Subsistema de Lógica del Negocio

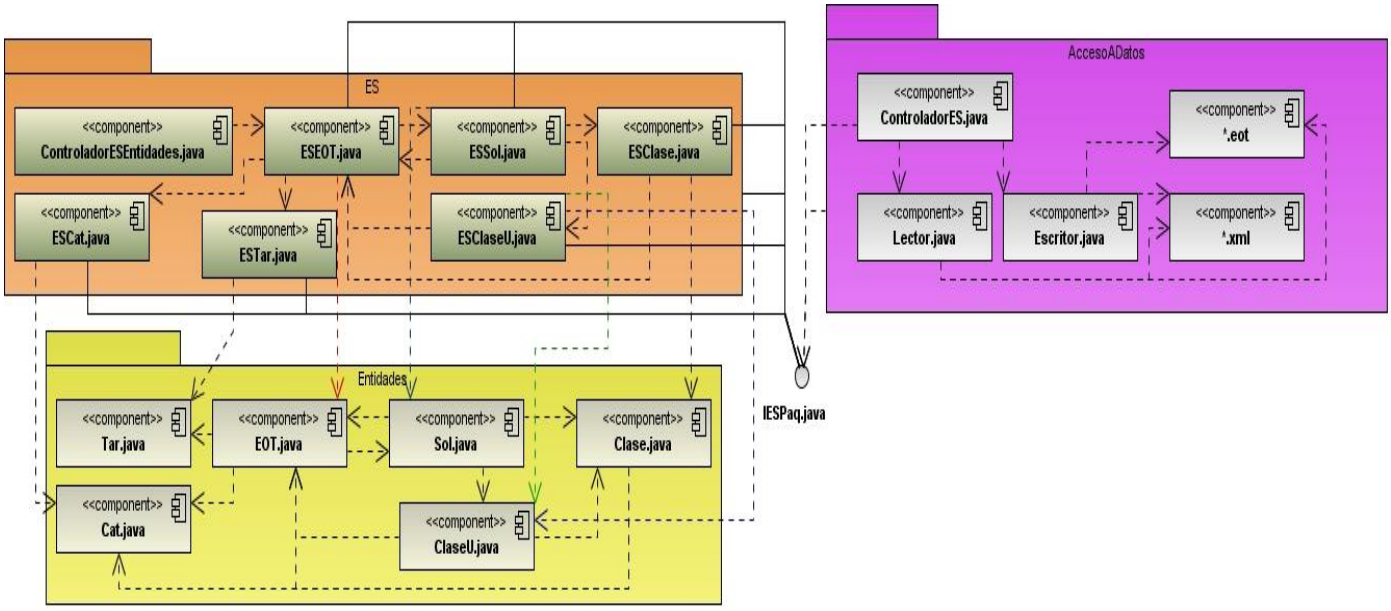


Fig. 25. Subsistema Entrada y Salida y su relación con los subsistemas Acceso a Datos y Entidades.

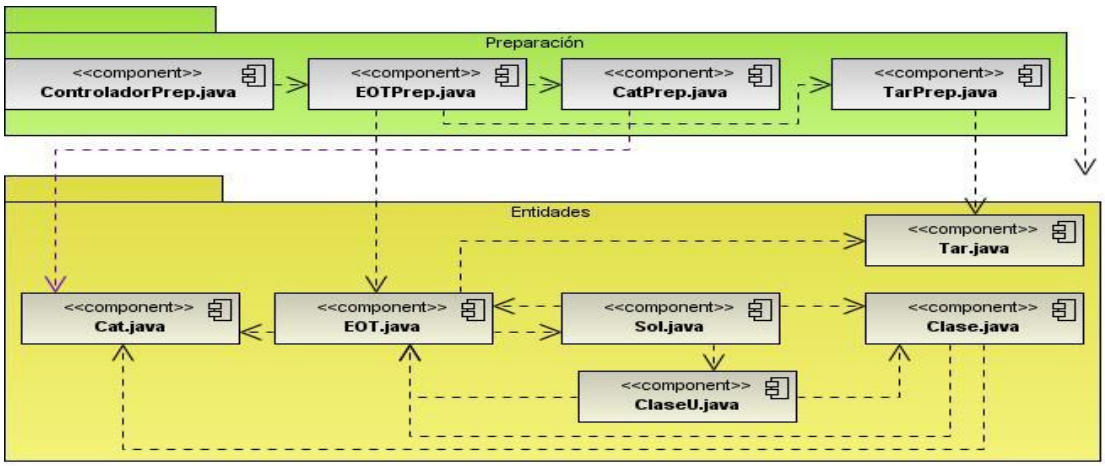


Fig. 26. Subsistema Preparación y su relación con el subsistema Entidades.

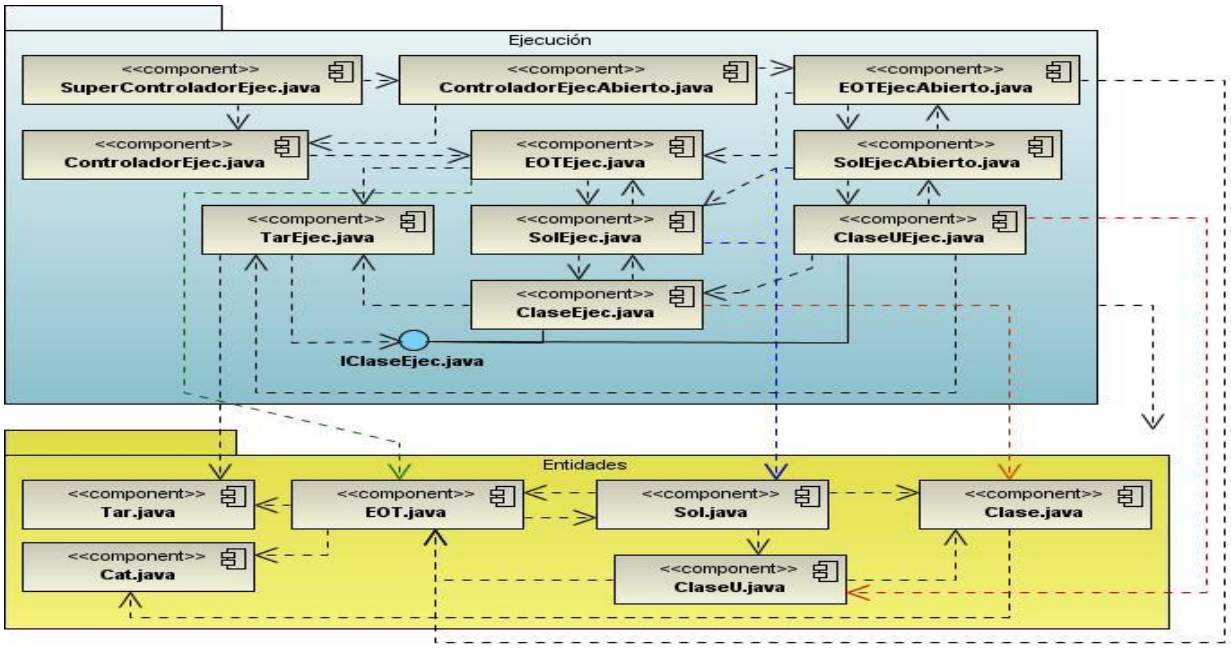


Fig. 27. Subsistema Ejecución y su relación con el subsistema Entidades

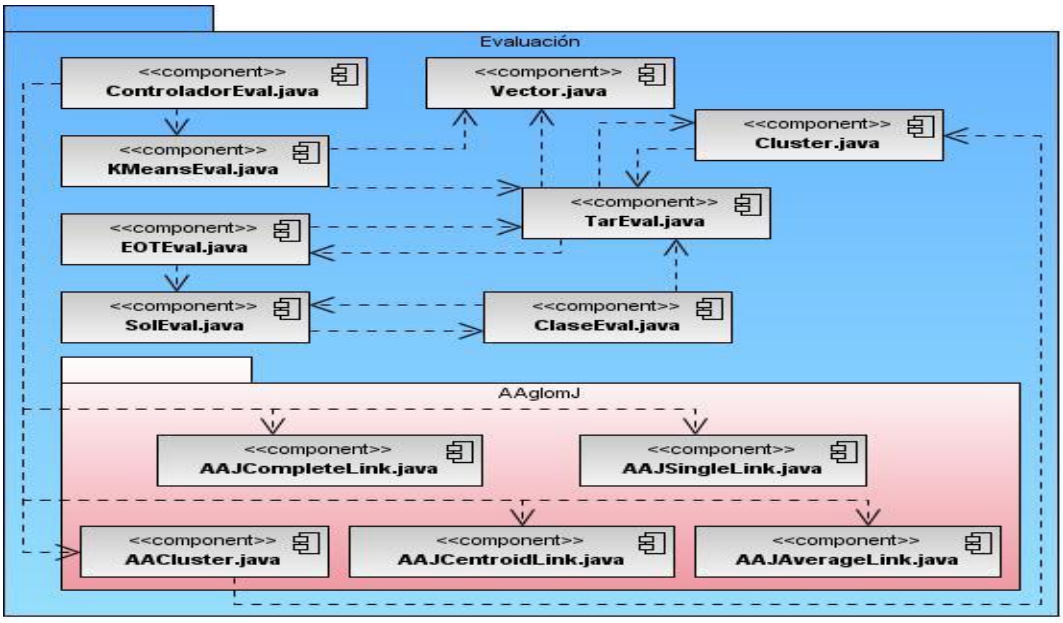
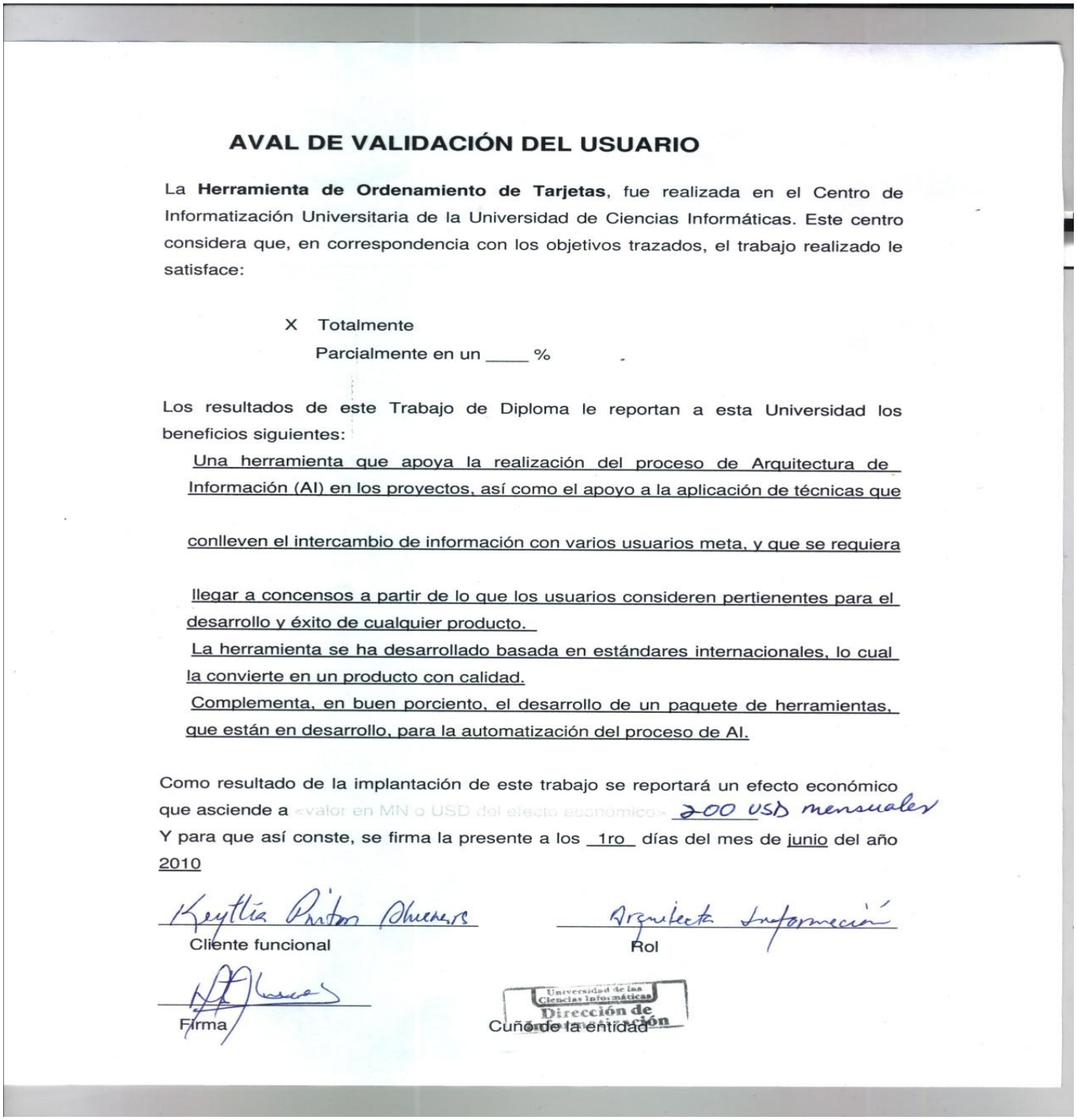


Fig. 28. Subsistema Evaluación y Subsistema Analizador de Aglomerados Jerárquicos.



AVAL DE VALIDACIÓN DEL USUARIO

La **Herramienta de Ordenamiento de Tarjetas**, fue realizada en el Centro de Informatización Universitaria de la Universidad de Ciencias Informáticas. Este centro considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface:

X Totalmente

Parcialmente en un _____ %

Los resultados de este Trabajo de Diploma le reportan a esta Universidad los beneficios siguientes:

Una herramienta que apoya la realización del proceso de Arquitectura de Información (AI) en los proyectos, así como el apoyo a la aplicación de técnicas que conlleven el intercambio de información con varios usuarios meta, y que se requiera llegar a concensos a partir de lo que los usuarios consideren pertienentes para el desarrollo y éxito de cualquier producto.

La herramienta se ha desarrollado basada en estándares internacionales, lo cual la convierte en un producto con calidad.

Complementa, en buen porciento, el desarrollo de un paquete de herramientas, que están en desarrollo, para la automatización del proceso de AI.

Como resultado de la implantación de este trabajo se reportará un efecto económico que asciende a <valor en MN o USD del efecto económico> 200 USD mensuales

Y para que así conste, se firma la presente a los 1ro días del mes de junio del año 2010

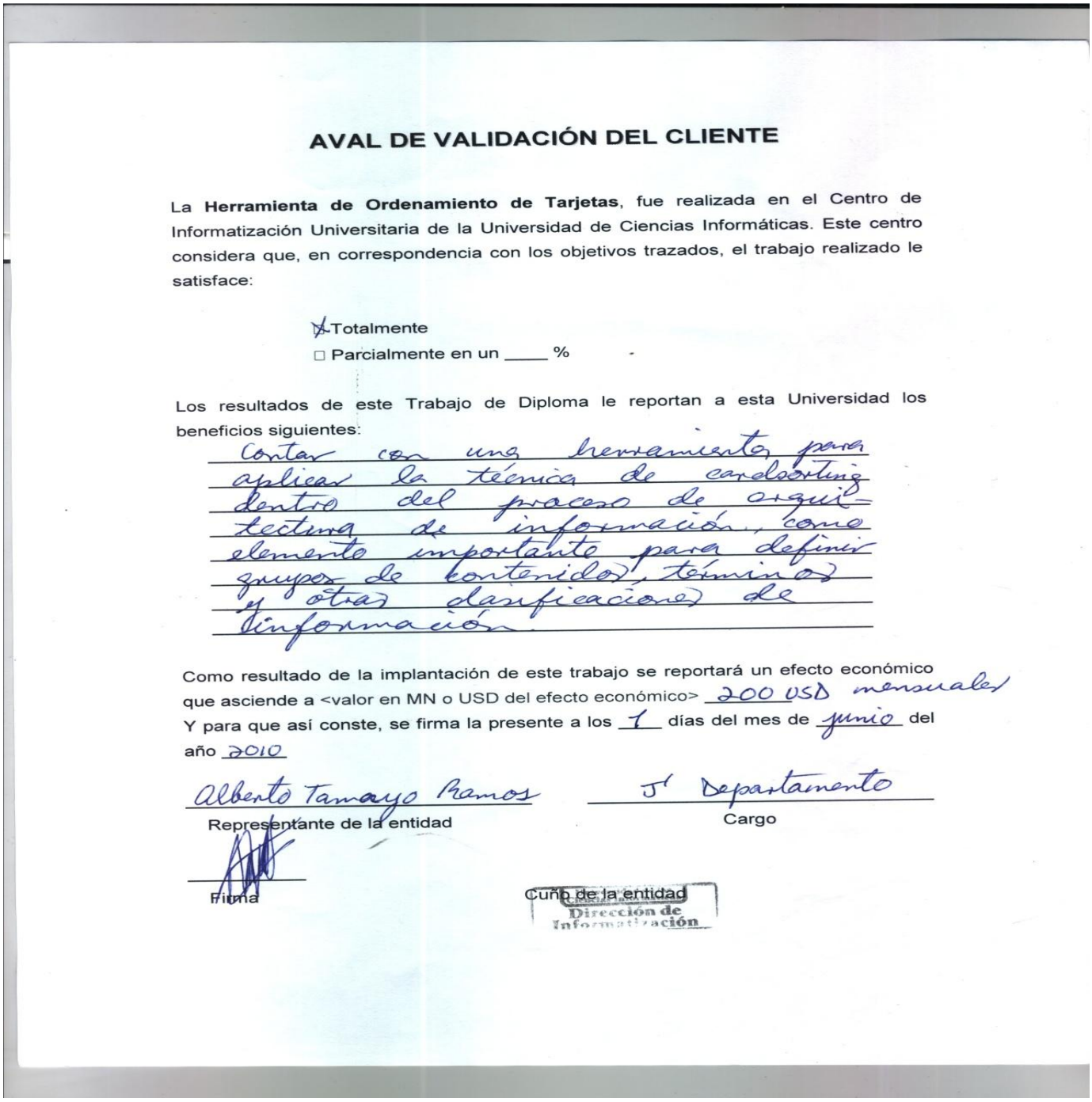
Keytlia Prieto Chenard — Cliente funcional

Arquitecta Información — Rol

Firma

Universidad de las Ciencias Informáticas Dirección de Informatización — Cuño de la entidad

Fig. 29. Aval del cliente.



AVAL DE VALIDACIÓN DEL CLIENTE

La **Herramienta de Ordenamiento de Tarjetas**, fue realizada en el Centro de Informatización Universitaria de la Universidad de Ciencias Informáticas. Este centro considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface:

☒ Totalmente
☐ Parcialmente en un ____ %

Los resultados de este Trabajo de Diploma le reportan a esta Universidad los beneficios siguientes:

Contar con una herramienta para aplicar la técnica de cardsorting dentro del proceso de arquitectura de información, como elemento importante para definir grupos de contenidos, términos y otras clasificaciones de información.

Como resultado de la implantación de este trabajo se reportará un efecto económico que asciende a <valor en MN o USD del efecto económico> 200 USD mensuales
Y para que así conste, se firma la presente a los 1 días del mes de junio del año 2010

Alberto Tamayo Ramos — Representante de la entidad

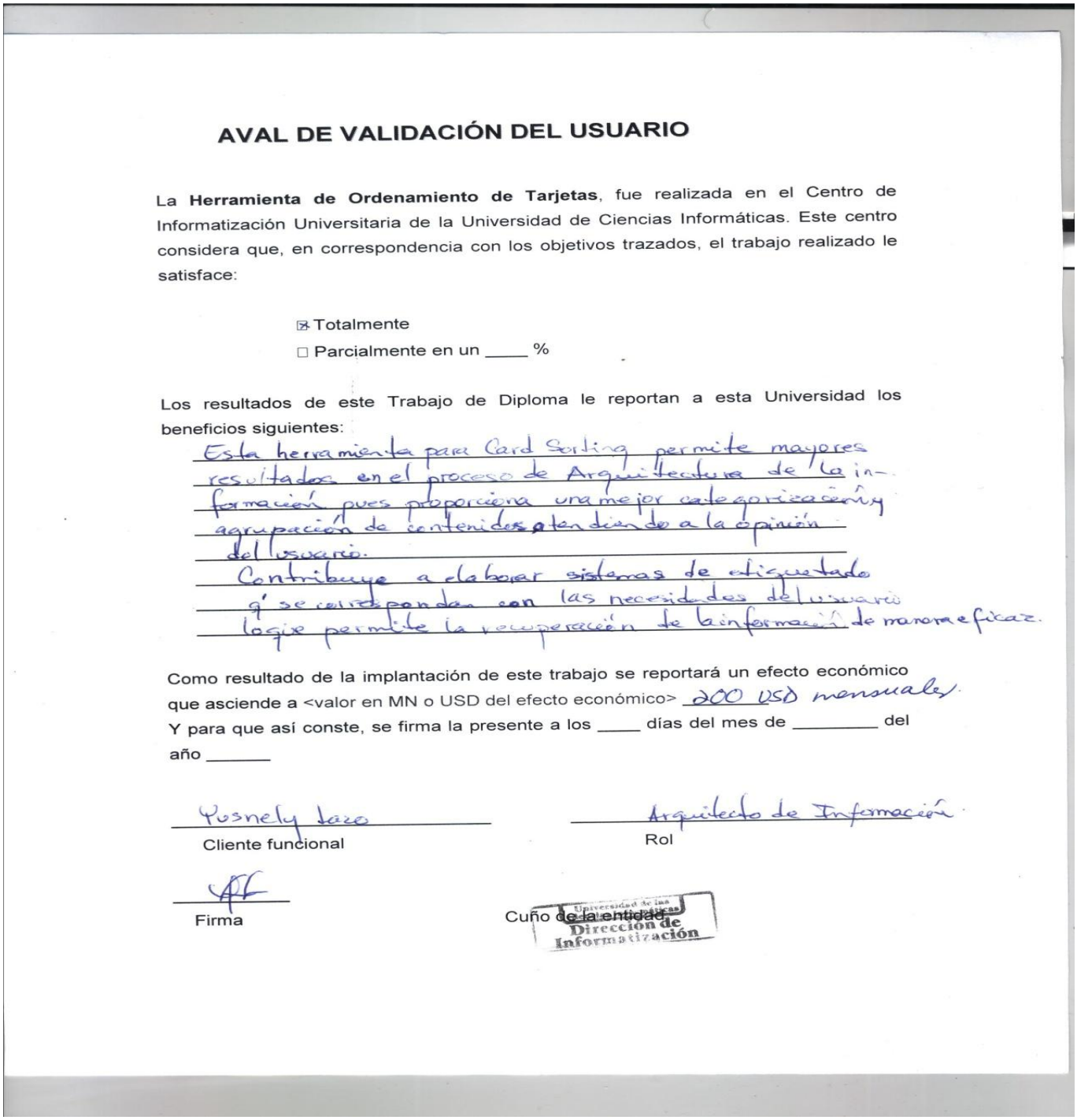
1ª Departamento — Cargo

Firma

Cuño de la entidad

Dirección de Informatización

Fig. 30. Aval de la entidad.



AVAL DE VALIDACIÓN DEL USUARIO

La **Herramienta de Ordenamiento de Tarjetas**, fue realizada en el Centro de Informatización Universitaria de la Universidad de Ciencias Informáticas. Este centro considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface:

☒ Totalmente
☐ Parcialmente en un _____ %

Los resultados de este Trabajo de Diploma le reportan a esta Universidad los beneficios siguientes:

Esta herramienta para Card Sorting permite mayores resultados en el proceso de Arquitectura de la información pues proporciona una mejor categorización y agrupación de contenidos atendiendo a la opinión del usuario.
Contribuye a elaborar sistemas de etiquetado q se correspondan con las necesidades del usuario lógico permite la recuperación de la información de manera eficaz.

Como resultado de la implantación de este trabajo se reportará un efecto económico que asciende a <valor en MN o USD del efecto económico> 200 USD mensuales.
Y para que así conste, se firma la presente a los _____ días del mes de _____ del año _____

Yusnely Lazo
Cliente funcional

Arquitecto de Información
Rol

Firma

Cuño de la entidad

Universidad de las Ciencias Informáticas
Dirección de Informatización

Fig. 31. Aval del cliente.