Universidad de las Ciencias Informáticas

Facultad 1

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Título: Propuesta de herramienta en el paquete Abad para la aplicación de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia.

Autora: Gleydis María Rodríguez Ramírez

Tutores: Ing. Yanicet Aveleira Rodríguez
Ing. Dayaisis Bárbara Pompa

Habana, junio 2012

Frase

El problema no es si las máquinas piensan,

el problema es si los hombres lo hacen.

Federico II

Declaración de autoría

Declaro que soy la única autora de este trabajo y autorizo al Centro de Informatización Universitaria de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____

Gleydis María Rodríguez Ramírez

Autora

_____ _____

Ing. Yanicet Aveleira Rodríguez

Tutor

Ing. Dayaisis B. Bernis Pompa

Tutor

Datos de contacto

Ing. Yanicet Aveleira Rodríguez

Ingeniera en Ciencias Informáticas, graduada en 2008. Perteneció al grupo de trabajo Arquitectura y Estándares de información de la Dirección Técnica de la Universidad de las Ciencias Informáticas. Actualmente es líder del proyecto Paquete de herramientas para diseñar experiencia de usuario (Abad) del Centro de Informatización Universitaria (CENIA) de la facultad 1 de la propia universidad. Profesor instructor.

Correo: yaveleira@uci.cu

Ing. Dayaisis Bárbara Bernis Pompa

Ingeniera en Ciencias Informáticas, graduada en 2009. Perteneció al grupo de trabajo Arquitectura y Estándares de información de la Dirección Técnica de la Universidad de las Ciencias Informáticas Trabajó como analista principal del proyecto Paquete de herramientas para diseñar experiencia de usuario (Abad) del Centro de Informatización Universitaria (CENIA) de la facultad 1 de la propia universidad. Actualmente desempeña el mismo rol en proyecto Identificación y Control de Acceso. Imparte en pregrado la asignatura Arquitectura de Información.

Correo: dbbernis@uci.cu

Agradecimientos

Es difícil agradecer en tan poco espacio a tantas personas involucradas en la realización de este sueño, pues es correr el riesgo de olvidar alguna, aunque espero no hacerlo y si lo hago pido perdón porque realmente son muy importantes para mí. Agradezco a Fidel Castro Ruz y a la Revolución cubana por darme la posibilidad de desfilar en esta gran tropa de futuro.

Vamos a empezar por el árbol genealógico pues sin dudas sin ellos definitivamente no estuviera aquí:

- A mami y mis abuelos que aunque algunos no están físicamente hoy es un día de orgullo para ellos donde quiera que se encuentren.
- A mi mamá y a mi papá, que decirles si ellos son mi razón de ser y existir, gracias a ellos que nunca han descansado en la meta de hacer de mí una mejor persona, ojalá y la vida me permita retribuirles todo ese amor y cariño. Estoy orgullosa de tener la mejor madre y el mejor padre del mundo.
- A Frank “mi jevito” que aunque llegó a mi vida mucho después de lo que hubiera deseado se ha convertido en un elemento principal para vivir, gracias por ser tú, por aceptarme como soy, por comprenderme y sobre todo por tolerarme.
- A mis hermanos y a mi familia que siempre me han apoyado y dado fuerzas para seguir adelante.
- A mis suegros y a mi cuñada por acogerme y hacerme miembro de su familia, para mí es un honor.
- A los buenos amigos, los que no llevan nombres, porque saben que en mi corazón ocupan un lugar especial y que aunque la vida me separa de algunos siempre los recordaré.
- A mis tutoras que me han ayudado y soportado durante estos 3 años, más que tutoras han sido mis amigas, gracias por permitirme burlarme de mis amigas al recordarles que yo si he tenido tutoras geniales y ellas no.
- A yani que durante los 11 meses de misión se convirtió en mi amiga, mi hermana y a veces hasta en mi madre para aconsejarme y cuidarme como la mejor de todas. Un beso grande mi amiga.

A todos gracias y mil gracias por ayudarme y acompañarme en esta travesía. La vida nos regala cosas lindas a menudo, pero a mí me ha premiado con creces al permitirme contar con personas como ustedes.

Dedicatoria

A mi mamá, mi papá, mi abuela y mi jevito.

Resumen

La presente investigación describe los aspectos teóricos conceptuales relacionados con la técnica de ordenamiento de tarjetas (OT) (conocido en inglés como *Card sorting*) y el método de análisis de secuencia (AS) en la arquitectura de la información (AI). Se realiza un estudio de las soluciones similares que existen, para conocer cómo se aplican en el mundo. A partir de este análisis el presente trabajo propone la modelación de una herramienta que integre la técnica de ordenamiento de tarjetas y el método de análisis de secuencia, que apoye con sus resultados la toma de decisiones de los profesionales de la AI en la representación de los contenidos en las interfaces de usuario. El desarrollo de la herramienta será guiado por la metodología de desarrollo SXP, como lenguaje de modelado UML, herramienta de diagramación Visual Parading y la herramienta Pencil para el prototipado.

Palabras claves: Análisis de secuencia, Arquitectura de información, Ordenamiento de tarjetas.

Índice

Índice

Índice de tablas y figuras

Tablas

Figuras

Introducción

En los últimos años se ha incrementado el interés por perfeccionar las nuevas tecnologías de la información y las comunicaciones (NTICs) y para ello se han incorporado una serie de buenas prácticas dentro del proceso de desarrollo de *software* con el objetivo de lograr que los productos cumplan con las expectativas de sus usuarios finales. En la búsqueda de soluciones de diseño más integradoras, se ha popularizado la "Experiencia del Usuario" como un nuevo enfoque para el desarrollo de productos interactivos. Incorporar esta iniciativa al proceso de desarrollo de *software* permite mejorar la calidad de los productos y centrarse verdaderamente en la satisfacción del usuario final. En el diseño de la experiencia de usuario se involucran varias disciplinas como la usabilidad, diseño de interacción, accesibilidad y la arquitectura de información entre otras.

La AI debe ser entendida como la disciplina (arte y ciencia) que proporciona métodos y herramientas para estructurar, organizar y etiquetar los componentes que conforman los entornos informacionales. En este sentido, persigue el objetivo de facilitar el acceso a la información contenida en esos entornos y mejorar, así, su utilidad y aprovechamiento por parte de sus usuarios (Pérez, Montoro, 2010).

En la AI se aplican varias técnicas clasificadas por sus características en grupos entre las que se encuentran las técnicas matemáticas (co-ocurrencia) que consisten en la aplicación de análisis para cuantificar resultados, hacer precisa la toma de decisiones y obtener del usuario la información necesaria para mostrar en un producto final.

Dentro de las técnicas matemáticas se encuentran la de ordenamiento de tarjetas, que junto al método de análisis de secuencia brindan al arquitecto de información[1] un resultado más exacto de la representación de los contenidos en las aplicaciones. El objetivo en el primer caso es definir cuáles serán las agrupaciones de las etiquetas y los contenidos y en el segundo especificar el orden que tendrán, complementándose así la técnica de ordenamiento de tarjetas con el método de análisis de secuencia.

La técnica de ordenamiento de tarjetas es una de las más utilizadas por los arquitectos de información. La misma se basa en la organización de contenidos centrada en el criterio de agrupamiento de los usuarios finales de los productos, que busca facilitar el acceso y la navegación a través de las

[1] Persona que crea el mapa o la estructura de información que permite a otros encontrar su camino personal al conocimiento" (Wurman, 1997).

aplicaciones. El análisis de secuencia es un método muy similar a la técnica anterior. La diferencia radica en que los resultados tienen otro objetivo: formar una secuencia de los elementos para ser usada en el producto final.

En la actualidad es muy común que se utilice la técnica de ordenamiento de tarjetas para la toma de decisiones a la hora de confeccionar un sitio web o una aplicación de escritorio. La Universidad de las Ciencias Informáticas (UCI) no está ajena a estos avances en la AI y también hace suya la aplicación de esta técnica en la AI que realiza a los diferentes productos de *software*.

A finales de febrero del 2010 la UCI creó el proyecto Abad con el objetivo de desarrollar una solución integrada para la gestión de los flujos de AI en los proyectos de desarrollo de soluciones informáticas. En este mismo año dicho proyecto realizó un estudio de los diferentes trabajos existentes en la universidad referentes al tema de AI y la técnica de ordenamiento de tarjetas y basado en los de análisis y diseño de OT encontrados desarrolló una aplicación llamada K-sort para la automatización de esta técnica.

Esta aplicación no cumplió con las expectativas esperadas ni logró brindar al arquitecto de información el apoyo necesario en la toma de decisiones para la representación de contenidos pues, desde su análisis y diseño no tuvo en cuenta elementos fundamentales que fortalecen la interpretación del modelo mental[2] de los usuarios, como son:

- El análisis y diseño del método de análisis de secuencia.
- La estructuración de contenidos dentro de los prototipos propuestos, no permitía completar el flujo de los procesos que implican la realización de la técnica de ordenamiento de tarjetas.
- La mezcla varias soluciones en casos de contar con más de un participante en la ejecución de un ejercicio.
- La interfaz no tenía una estructuración adecuada de la información, dificultando el trabajo de los usuarios con la misma.
- Los procesos de los participantes[3] estaban concebidos junto a los del arquitecto de información.

[2] Representaciones internas de cómo los usuarios imaginan la organización de la información.
[3] Segmento de la audiencia del producto final.

- La automatización con calidad de todos los procesos de la técnica de ordenamiento de tarjetas.

Teniendo en cuenta lo antes expuesto se plantea como **problema de la investigación**: ¿Cómo apoyar la toma de decisiones del arquitecto de información en la representación de contenidos a partir de la aplicación de la técnica de ordenamiento de tarjetas y el método análisis de secuencia?

Se define como **objeto de estudio**: el proceso de arquitectura de información.

El campo de acción: la técnica de ordenamiento de tarjetas y el método de análisis de secuencia en el proceso de arquitectura de información.

Se plantea como **objetivo general**: Modelar una herramienta informática en el proyecto Abad para apoyar la toma de decisiones del arquitecto de información en la representación de contenidos, mediante la incorporación de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia.

Como **objetivos específicos** se plantean:

- Caracterizar la técnica de ordenamiento de tarjetas y el método de análisis de secuencia, dentro de la arquitectura de información.
- Caracterizar la metodología de desarrollo, tecnologías y lenguajes de programación para el desarrollo del *software*.
- Definir las características del sistema propuesto.
- Describir la arquitectura del sistema propuesto.

El cumplimiento efectivo del objetivo trazado dependerá de las siguientes **tareas de la investigación**:

- Análisis del estado del arte de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia.
- Análisis de soluciones informáticas existentes que incluyan el desarrollo de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia.
- Identificación de las herramientas a utilizar para el diseño e implementación de la propuesta.
- Identificación de los algoritmos a usar para la evaluación de los ejercicios.
- Descripción de los procesos del negocio.
- Identificación de los requisitos que el sistema debe cumplir.

- Descripción de las historias de usuarios.
- Descripción de la arquitectura del *software*.
- Confección del prototipo no funcional.
- Definición de los patrones de diseño a utilizar.
- Definición de los casos de pruebas basados en las historias de usuarios.

Posible resultado de la investigación: La modelación de una herramienta informática que apoye la toma de decisiones del arquitecto de información para la representación de contenidos.

Como **métodos teóricos** se utilizaron:

Análisis histórico – lógico: se utilizó en el estudio y análisis de las bibliografías relacionadas con el tema para obtener elementos teóricos relacionados con el objeto de investigación.

Analítico-Sintético: se empleó para el estudio y profundización de la evolución de las aplicaciones que automatizan la técnica de ordenamiento de tarjetas, lo que facilitó la indagación de soluciones al problema planteado.

Modelación: Se crearon modelos que permitieron representar las características de la propuesta de solución.

Antecedentes

La presente investigación tiene sus antecedentes en los trabajos de diploma: “Análisis de un sistema automatizado a través de la técnica de Card sorting u Ordenación de Tarjetas” (Sola Padilla, 2008), “Análisis y Diseño de un sistema para la aplicación de técnicas de Card sorting en la obtención de Arquitecturas de Información” (Orovio Pino, 2009), “Implementación del módulo “Técnica de ordenamiento de tarjetas” para la plataforma de arquitectura de la información “ABAD”(Espinosa Ramírez, Jiménez Morales 2010) y en las aplicaciones WebSort (Lime & Chile, 2009), Optimal Sort (Optimal Sort, 2009), SynCaps (Syntagm, 1995) y UXsort (UsabilityCentric, 2009).

Estructuración del informe:

EL presente informe está compuesto por un resumen, una introducción, tres capítulos, conclusiones, recomendaciones, bibliografía y los anexos.

El **capítulo 1** incluye una descripción teórica de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia, un estudio del estado del arte relacionado con la técnica de ordenamiento de tarjetas y el método de análisis de secuencia. Así como, la fundamentación teórica de la necesidad de realizar la automatización de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia para la plataforma de arquitectura de información “Abad” y de los elementos técnicos para desarrollar la solución.

El **capítulo 2** incluye una descripción de las reglas del negocio, la definición de los procesos y capacidades o condiciones que la solución debe cumplir así como, una descripción de las historias de usuario y del sistema propuesto.

El **capítulo 3** presenta la validación de los requisitos capturados para la propuesta de solución, los patrones de diseño y su fundamentación, así como una descripción de la propuesta de la arquitectura de *software* y paquetes de la solución a desarrollar. Presenta además, la validación de los métodos matemáticos propuestos para la evaluación de la técnica de OT y el diseño de los casos de prueba.

Capítulo 1 Fundamentación teórica

1.1 Introducción

En este capítulo se exponen los aspectos generales de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia. Se realiza un estudio del estado del arte de cómo se aplica la técnica de ordenamiento de tarjetas y el método de análisis de secuencia, así como de los sistemas que realizan funciones similares a la solución que se propone en este trabajo. Además, de realizarse una crítica valoración de la metodología, el lenguaje de programación y las herramientas propuestas para el desarrollo de la aplicación.

1.2 Conceptos asociados al dominio del problema

- **Análisis cualitativo:** En el análisis cualitativo se observan qué términos ofrecieron dificultad, cuál no se comprendió, cuál es considerado el primero y porqué, qué criterio de organización usó el usuario, qué otro término propone algún usuario, cuál término resulta ambiguo (Ronda[4], Mesa, 2005).

- **Análisis cuantitativo:** Se tabulan los resultados y se realiza el análisis de co-ocurrencia, luego se grafica (Ronda, Mesa, 2005).

- **Análisis de co-ocurrencia:** El análisis de co-ocurrencia consiste en el estudio de la aparición conjunta de dos o más etiquetas y/o grupo de etiquetas. Es decir, se analiza cuántas veces se repiten los distintos criterios de secuencialización que establece cada usuario que realizó la prueba (Ronda, Mesa, 2005).

- **Organizar secuencialmente:** El orden a dar a las etiquetas; de forma tal que represente de la mejor manera posible las estructuras mentales de los usuarios finales. Las formas de organización secuencial más conocidas son: la numérica (1, 2, 3,4…), la alfabética (a, b, c…) o la cronológica (1954, 1955, 1956…) (Ronda, Mesa, 2005).

[4] Rodrigo Ronda León: Licenciado en Bibliotecología y Ciencia de la Información de la Facultad de Comunicación de la Universidad de la Habana. Ha trabajado como arquitecto de información, analista de sistema y diseñador de información en instituciones cubanas. Profesor adjunto de la Facultad de Comunicación de la Universidad de la Habana en la carrera Bibliotecología y Ciencia de la Información. Cliente funcional de la propuesta de solución y consultor del tema de AI en el centro, por estas razones se toma como referente teórico en esta investigación.

- **Dendogramas:** Es un tipo de representación gráfica de datos en forma de árbol (Dendro=árbol) que organiza los datos en subcategorías que se van dividiendo en otros hasta llegar al nivel de detalle deseado (Vicente, 2004).

- **Clústeres:** Agrupamientos (Vicente, 2004).

1.3 Ordenamiento de tarjetas

El OT es una técnica centrada en el usuario, utilizada por los arquitectos de la información 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 puede ser de dos tipos: abierto[5] o cerrado[6]"; y está compuesto por tres fases: preparación, ejecución y evaluación. En la primera fase se prepara el ejercicio de ordenamiento de tarjetas por parte del arquitecto de información mediante la elaboración de las instrucciones, las tarjetas[7] y las categorías[8]. Durante la fase de ejecución se les ofrecen las tarjetas y categorías a los participantes para que las agrupen en clases[9] según su criterio personal. En la última fase de la técnica se evalúan las clases obtenidas mediante algoritmos de formación de agrupamientos, obteniéndose aportes para el diseño y organización de la información de acuerdo con los patrones mentales de los participantes. Para la evaluación de los ejercicios de ordenamiento de tarjetas se transforman las clases y tarjetas en vectores a partir de las ocurrencias de las tarjetas en cada uno de los agrupamientos, luego estos vectores se transforman en vectores distancia a través del método de distancia euclidiana[10], estos vectores distancia son utilizados por los algoritmos de formación de agrupamientos planos y jerárquicos.

El algoritmo de formación de agrupamiento plano (ω) que se utilizará es el K – Means, cuyo objetivo es minimizar el cuadrado de la distancia euclidiana de los elementos del agrupamiento (vectores) al

[5] Los participantes pueden crear las categorías de acuerdo a su criterio personal.
[6] Las categorías son creadas por el arquitecto de información y no pueden ser modificadas por el usuario, aunque estos pueden aportar sugerencias.
[7] Cada tarjeta contiene un término y un concepto relacionado.
[8] Espacio utilizado por los participantes para crear las clases durante la ejecución.
[9] Agrupamiento de tarjetas creado por un participante sobre una categoría.
[10] Proximidad entre elementos.

centro del mismo. Este centro es definido como la media o el centroide ($\vec{\mu}$) y está determinado por la siguiente ecuación (Manning, 2007):

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

($\vec{\mu}$): Mediana

(ω): Clúster

($\vec{x}$): Vectores

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 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: suele indicar que se ha obtenido un buen agrupamiento, a excepción de los casos en los que se obtienen mínimos locales; 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.

Los algoritmos jerárquicos producen una secuencia anidada de particiones del conjunto de objetos, es decir, los grupos se organizan de forma jerárquica y cada clúster puede verse como la unión de otros clústeres, obteniendo así distintos niveles de jerarquía de grupos. Son basados en la distancia euclidiana. Comienzan considerando a cada objeto como grupos unitarios y en cada iteración se unen los dos grupos más cercanos hasta que se obtenga un único grupo o hasta que se cumpla un criterio

de parada determinado. Los criterios de parada más empleados son: cuando se obtengan los grupos predefinidos o cuando la distancia entre el par de grupos más cercanos sea mayor que la mínima obtenida (Vicente, 2004).Los algoritmos de formación de agrupamientos jerárquicos que se utilizarán son:

- *Single–Link*: En cada paso se unen los dos grupos cuyos elementos tienen la mínima distancia.
- *Complete–Link:* En cada paso se unen los dos grupos tal que su unión tiene el diámetro mínimo o los dos grupos con la menor distancia entre sus elementos.
- *Group–Average:* En cada paso se unen los dos grupos tal que tienen la mínima distancia promedio entre sus puntos.

La diferencia de estos algoritmos consiste en el criterio de similitud que se toma para formar los agrupamientos.

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.

Tabla 1. Criterios de similitud para algoritmos jerárquicos.

La técnica de ordenamiento de tarjetas presenta ventajas y desventajas que se deben tener presentes para determinar la forma en que esta se aplica:

Ventajas:

- Simple: El ordenamiento de tarjetas es simple para el organizador y los participantes.
- 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.

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 encaminarse hacia una estructura de información que no sea la más indicada ante los roles que deben desempeñar los usuarios.

- Los resultados pueden variar: El ordenamiento de tarjetas puede proveer aleatoriamente resultados similares entre los participantes, o estos pueden ser muy diferentes entre sí.

Muchas son las instituciones que utilizan la técnica de ordenamiento de tarjetas para el diseño de sus portales web, en busca de establecer patrones o modelos mentales de organización de contenidos por parte de los usuarios que los mismos utilizan. En la UCI se encuentra la aplicación K-sort desarrollada por el proyecto Abad para el ordenamiento de tarjetas. La misma busca ampliar sus funcionalidades para lograr modelos más exactos de organización y para ello se pretende implementar en la propuesta de solución como complemento de la técnica de ordenamiento de tarjetas el análisis de secuencia.

1.4 Análisis de secuencia

Es un método que "consiste en la realización de una serie de pruebas a usuarios potenciales del producto, y el posterior análisis cualitativo y cuantitativo de esos resultados, para ayudar a definir la secuencia de las etiquetas en el producto electrónico"(Ronda, Mesa, 2005).

El método de análisis de secuencia tiene una gran similitud en cuanto a forma de realización y análisis de los resultados, con la técnica de ordenamiento de tarjetas. Lo que cambia es la finalidad de los resultados. En la técnica de ordenamiento de tarjetas el objetivo es definir cuáles serán las agrupaciones de las etiquetas [11] y en el análisis de secuencia se define el orden que van a tener las mismas.

El análisis de secuencia puede ser combinado con otras técnicas tradicionales como la entrevista, la prueba de usuarios y la técnica de ordenamiento de tarjetas; lográndose así una mayor calidad y precisión en los resultados.

[11] Son las categorías en la técnica de ordenamiento de tarjetas y que pasan a llamarse etiquetas en el análisis de secuencia.

Para realizar en al análisis de secuencia se debe tener en cuenta (Ronda, Mesa, 2005)***:***

- Definición de los términos (si son de una barra de navegación, o un menú desplegable o una pequeña lista de términos). En este paso se aconsejan basarse en las técnicas de entrevista y encuesta del diseño centrado en el usuario.

- Confección de etiquetas con cada término acordado durante la realización del paso anterior. Cada etiqueta debe tener escrito un número en la parte posterior, con el objetivo de facilitarle el ordenamiento al arquitecto de información y poderlo tabular para realizar el análisis cuantitativo.

- Entrega del grupo de tarjetas a una muestra de usuarios potenciales y solicitarle a los mismos que las organicen consecutivamente por posiciones (1, 2, 3…) según su criterio individual.
- Realización del análisis cualitativo de la prueba.
- Recogida de los resultados y análisis cuantitativo.
- Conclusiones. Se define el orden secuencial, basándose en el análisis cualitativo y cuantitativo, y siguiendo los principios lógicos de ordenamiento.

1.5 Análisis de soluciones existentes

1.5.1 Web Sort

Es una herramienta web, la primera herramienta de ordenamiento de tarjeta en línea y todavía sigue mejorándose. Con un costo total de $79 por estudio más los descuentos. *WebSort* requiere el reproductor de Flash 9. Es utilizado por Yahoo, Paypal, Oracle entre otros. (Lime & Chile, 2009). Con esta herramienta se puede*:*

- Validar la estructura de su organización.
- Conseguir la regeneración en la intranet de su empresa.
- Aprender dónde agregar nuevo contenido a un sitio existente.
- Reasignar tareas y funciones dentro de un equipo.
- Crear flujos de trabajo viables para la gestión de proyectos.
- Organizar los productos en su tienda de comercio electrónico.
- Obtener las características de prioridad para el *software*.

1.5.2 OptimalSort

Es una herramienta web realizada por los desarrolladores del *OptimalWorkshop* que realizaron también herramientas *UX* remotas. Con un costo total de $109 por mes con estudios ilimitados durante ese tiempo. También tiene descuentos para más herramientas. Permite crear tarjetas para los

participantes del grupo en categorías que tengan sentido para ellos. Además, saber si los miembros del personal de desarrollo de las tarjetas son diferentes de los clientes a través de la opción crear encuesta. Entre sus múltiples opciones se encuentra la de personalizar el color de la página y el texto de instrucciones, permite generar una URL de la encuesta para publicarla en línea y luego recoger los resultados, una similitud de matriz para el análisis de pares ordenados, dos dendogramas de análisis de conglomerados visuales, una nueva hoja de cálculo de todos los análisis de datos avanzados (OptimalSort, 2009).

Tanto el *WebSort* como el *OptimalSort* son aplicaciones privativas y con alto costo de adquisición 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, además de no contar con el análisis de secuencia integrado.

1.5.3 Xsort

Es una aplicación web libre de ordenamiento de tarjetas diseñada para Mac OS X. Se centra en los usuarios de experiencia y en los científicos sociales (Xsort, 2011). Con esta herramienta se puede:

- Crear un entorno visual que simula una mesa con cartas.
- Apoyar el tipo de ejercicio abierto, semiabierto y cerrado.
- Controlar todos los aspectos del ejercicio (tipo de clasificación, la colocación de tarjetas, etc.).
- Mostrar los resultados estadísticos (árbol de racimo, tabla de distancias, etc.) al día en tiempo real.
- Mostrar de forma individual toda la información relacionada con una sesión.
- Seleccionar fácilmente las sesiones que desea utilizar sobre la base de criterios diferentes.
- Facilitar el bloqueo del documento para que los participantes puedan hacer una sola sesión.
- Está completamente integrado con Mac OS X (Intel y Macs basados en PowerPC).

El *Xsort* fue creado haciendo uso del lenguaje de programación orientado a objetos C# y del marco de trabajo Microsoft .net, por lo que no es compatible con la plataforma Abad.

1.5.4 Optimal Workshop

Es una herramienta web de pago, dispone de la posibilidad de dar de alta una cuenta totalmente gratuita. La limitación que presenta esta herramienta es que solo permite tener activos 3 proyectos y un límite de tarjetas de 30 por proyecto con un máximo de 10 participantes. Permite *Card sorting* con

OptimalSort, Árbol de pruebas con *Treejack*[12], primera impresión de prueba con *Chalkmark*, a distancia, sin moderador pruebas con usuarios. La herramienta se elabora a través de una serie de pantallas en las que se va confeccionando el *test* que luego será enviado a los usuarios. Dispone de plantillas de introducción y del detalle del proceso para informar a los participantes. Posteriormente existe una pestaña para analizar los resultados y la posibilidad de exportarlos en formato Excel. Lo mejor de la herramienta es que permite que el *test* se realice a distancia, cosa que puede ser considerada por algunos como contraproducente, ya que se pierde la observación e iteración con los participantes. No se recomienda su uso por la limitación de proyectos activos y la cantidad de participantes que permite (Optimal Workshop, 2010).

1.5.5 CardSword

Está destinado a agilizar el proceso de llevar a cabo una OT. Para lograr eso, *CardSword* ofrece funcionalidades para ayudar a cada paso de un OT.

La aplicación consta de dos partes:

CardSword: Esta es la aplicación principal, donde se puede crear, editar y eliminar proyectos. Permite añadir, editar, eliminar grupos predefinidos (clasificación cerrada). Añadir, editar, eliminar los elementos que se ordenarán. Crear ficheros de configuración de CardSword: Zorro (*. CSZ). Evaluar los resultados de la carga de los archivos de OT realizadas por los usuarios utilizando *CardSword*: Zorro.

CardSword: Zorro: es la aplicación que utilizan los participantes que participan en el OT.

Muchas son las aplicaciones que existen en el mundo para la aplicación de la técnica de ordenamiento de tarjetas, la mayoría desarrolladas en aplicaciones web; ninguna de las encontradas posee el método de análisis de secuencia. El estudio realizado de las mismas permitió identificar:

- Cómo se aplica de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia en el mundo.

[12] Treejack pruebas de una jerarquía de contenido. A menudo, esto es simplemente la estructura del sitio, mapa o “estructura de árbol”.

- La necesidad de integrar la técnica de ordenamiento de tarjetas y el método de análisis de secuencia en una sola herramienta al no contar con una que automatice estos dos procesos en una sola herramienta.
- Que servía como punto de partida para el levantamiento de los requisitos.
- Nuevos elementos para el diseño de la herramienta.

1.6 Metodología de desarrollo

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.

Las metodologías ágiles o ligeras se caracterizan por la simplicidad de sus reglas, 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* e identifica al cliente como un miembro activo dentro del equipo de desarrollo (Peñalver, G., Meneses, A., García, S. 2010).

Entre sus principales características están:

- Un desarrollo iterativo, ágil, dinámico y muy flexible.
- El *software* que funciona, por encima de la documentación exhaustiva.
- La colaboración con el cliente, por encima de la negociación contractual.
- La respuesta al cambio, por encima del seguimiento de un plan.
- A los individuos y su interacción, por encima de los procesos y las herramientas.

Las metodologías ágiles están concebidas para proyectos donde los cambios en el desarrollo del sistema se produzcan con mucha frecuencia.

SXP

La metodología SXP toma características de Scrum[13] y XP[14].

[13] Metodología ágil de gestión de proyectos cuyo objetivo primordial es elevar al máximo la productividad de un equipo.
[14] Metodología ágil que para el éxito del desarrollo de un *software*, promueve 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.

Consta de 4 fases principales:

1. Planificación-Definición donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
2. Desarrollo, es donde se realiza la implementación del sistema hasta que esté listo para ser entregado.
3. Entrega, puesta en marcha.
4. Mantenimiento, donde se realiza el soporte para el cliente.

De cada una de ellas se despliegan 7 flujos de trabajo: concepción inicial, captura de requisitos, diseño con metáforas, implementación, prueba, entrega de la documentación, soporte e investigación, este último se utiliza por el equipo de desarrollo cuando sea necesario, es decir, es un flujo que se puede mover y utilizarlo en cualquier parte del ciclo de vida del proyecto (Peñalver, G., Meneses, A., García, S. 2010.)

Se escoge la metodología SXP para guiar el proceso de desarrollo de la aplicación ya que es la metodología que propone el centro donde se encuentra el proyecto Abad. La misma está especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, y permite además seguir de forma clara el avance del equipo de desarrollo por parte del cliente.

1.7 Tipo de aplicación

En la ingeniería de *software* se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de internet o de una intranet mediante un navegador. Mientras que las aplicaciones de escritorio no necesitan un navegador para el acceso a la aplicación sino la instalación del mismo en la PC.

Los usuarios finales de la aplicación K-sort no siempre podrán contar en su institución con una red de computadoras para la aplicación de la técnica de ordenamiento de tarjeta y el método de análisis de secuencia, por lo que la propuesta garantizará una alternativa más viable al ser desarrollada como una aplicación de escritorio, que podrá ser usada bajo cualquier circunstancia.

1.8 Lenguajes de programación

Java

Java constituye un lenguaje maduro, eficaz y muy versátil. Permite que los desarrolladores creen un sistema informático en una plataforma y lo ejecuten en otra distinta de la que se creó inicialmente. (Universidad Salamanca, 1999).

Java cuenta con una amplia documentación de librerías, es ideal para aplicaciones multiplataforma, tiene licencia GNU GPL[15] posee bibliotecas que hacen posible que el desarrollo de aplicaciones de escritorio sea dinámico y relativamente sencillo. La solución propuesta será integrada a la plataforma Abad, que está desarrollada en Java por lo que se propone este lenguaje para el desarrollo de la misma.

1.9 Entorno de desarrollo integrado (IDE)

Para el desarrollo del sistema se identificó el NetBeans 6.9 como IDE[16] para desarrollar la aplicación. NetBeans 6.9 es un paquete completo y enriquecido disponible para sistemas operativos como Windows, Mac, Linux, etc. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicaciones que permite a los desarrolladores crear rápidamente aplicaciones de escritorio y aplicaciones móviles utilizando la plataforma Java. Se propone realizar desarrollo del sistema propuesto sobre el IDE NetBeans 6.9 basado en las facilidades que ofrece para escribir y ejecutar programas escritos en Java. Es distribuido de manera libre, sin restricciones de uso.

1.10 Modelado de *software*

Visual Paradigm

Visual Paradigm (VP) ayuda al equipo de desarrollo a controlar el progreso del proyecto y brinda un medio de comunicación. 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 (Visual Paradigm, 2010**)**.

[15] Licencia publica general
[16] Integrated Development Environment (en español Entorno de desarrollo integrado)

Se propone utilizar Visual Paradigm porque es una herramienta que utiliza “UML” como lenguaje de modelado, es una herramienta de diseño que tiene compatibilidad con el sistema operativo Linux. Tiene la ventaja de soportar el ciclo de vida completo del desarrollo de *software* y posee disponibilidad en múltiples plataformas. Es la herramienta de modelado que propone el centro donde se integrará el paquete de soluciones Abad.

1.11 Lenguaje unificado de modelado

El lenguaje unificado de modelado (UML) es un lenguaje 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”. Está compuesto por diversos elementos gráficos que se combinan para conformar diagramas que permiten visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de *software* (Jacobson, et al., 2000).

Se propone trabajar en el modelado del trabajo con este lenguaje ya que incluye los conceptos necesarios para utilizar un proceso moderno iterativo, basado en construir una sólida arquitectura para resolver requisitos.

1.12 Notación para el modelado de procesos de negocio

Notación para el modelado de procesos de negocio (BPMN): es un estándar internacional de modelado de procesos. Es independiente de cualquier metodología de modelado de procesos. Crea un puente estandarizado para disminuir la brecha entre los procesos de negocio y la implementación de estos. Permite modelar los procesos de una manera unificada y estandarizada permitiendo un entendimiento a todas las personas de una organización (Owen, Raj, 2003). Se escoge BPMN como estándar de notación para el modelado de los procesos de negocio.

1.13 Herramienta para el prototipado del *software*

Pencil Project

Pencil Project está disponible para ordenadores con sistema operativo Windows o GNU/Linux. Es una solución muy versátil que ofrece una librería formada por elementos gráficos para el prototipado web,

que admiten una edición posterior y la incorporación de nuevos elementos gráficos externos. Permite la exportación a diferentes formatos como HTML, Png, *Open office*, Word y PDF.

Se escoge Pencil Project para el prototipado de la propuesta por ser la solución informática multiplataforma, gratuita y de código abierto más especializada para la realización de prototipos de interfaz de usuario.

1.14 Conclusiones del capítulo

Con la realización de este capítulo se concluye que:

- El estudio del estado del arte permite conocer cómo se aplica la técnica de ordenamiento de tarjetas y el método de análisis de secuencia en el mundo.
- La definición de los conceptos asociados al problema y los conceptos de ordenamiento de tarjetas y análisis de secuencia permiten un mejor entendimiento del negocio.
- El estudio de las herramientas homólogas a la que se propone permite demostrar la importancia del trabajo al integrar la técnica de ordenamiento de tarjetas y el método de análisis de secuencia en una sola herramienta, al no conocerse ninguna que integre estos procesos hasta el momento.
- La fundamentación de la metodología de desarrollo, los lenguajes y las herramientas propuestas permite un mejor desarrollo de la solución propuesta.

Capítulo 2 Características del sistema

2.1 Introducción

En este capítulo se expone una descripción del negocio. Para su mejor comprensión se describen los procesos del negocio, los requisitos del *software*, se realiza una descripción de las historias de usuarios y se describe la solución propuesta.

2.2 Descripción de la solución propuesta

Abad será un paquete de aplicaciones donde los profesionales de la UCI dedicados a la AI, podrán gestionar procesos que soportan la misma. Este trabajo propone los artefactos necesarios para la implementación paralela de la herramienta K-sort que permita la automatización de los procesos de preparación, ejecución y evaluación de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia. La herramienta será integrada al paquete de soluciones Abad, la misma permitirá crear tarjetas, categorías, etiquetas y posiciones para los ejercicios de ordenamiento de tarjetas y análisis de secuencia, mezclará diferentes soluciones y ofrecerá dendogramas y gráficos para su posterior interpretación por el arquitecto de información. Con la solución propuesta el arquitecto de información obtendrá un apoyo para interpretar el mapa mental de los usuarios.

2.3 Descripción del negocio

Es necesario comenzar por la modelación del negocio, en este caso se modelará por procesos. Este modelado consiste en describir la realidad, de manera que pueda ser entendida y de ser necesario modificada con el fin de incorporarle mejoras. Es importante contar con una notación que permita modelar con la mayor claridad posible la esencia del negocio, por lo que se utiliza la notación BPMN mencionada anteriormente en el capítulo 1.

Tanto en el ordenamiento de tarjetas como en el análisis de secuencia el arquitecto de información es el responsable directamente de la preparación, evaluación y control de estos procesos. En el caso de la ejecución solo es directamente responsable del control de la actividad aunque puede estar o no involucrado en la resolución del ejercicio.

El ordenamiento de tarjetas comienza una vez que el arquitecto de información decide aplicar esta técnica a una serie de usuarios del producto y desarrollar la arquitectura de información en dependencia de sus mapas mentales. El arquitecto prepara el ejercicio de ordenamiento de tarjetas en

dependencia de los elementos a evaluar y lo entrega a los participantes, quienes ejecutan el ejercicio. Finalmente, se tabulan los resultados y se emite una conclusión.

La técnica de ordenamiento de tarjetas cuenta con tres procesos fundamentales, los cuales serán descritos a continuación.

EOT: Ejercicio de ordenamiento de tarjeta.

- El proceso de preparación de un ejercicio de ordenamiento de tarjetas.

2.3.1 Preparación EOT

Ficha de Proceso	
Proceso:	Preparación de ordenamiento de tarjetas
Entradas:	Solicitud de aplicación del ejercicio.
Salidas:	Se obtiene un EOT preparado
Involucrados:	Arquitecto de la información.
Descripción	
Actividad 1 Organización de la información.	El arquitecto de la información realiza una solicitud de recomendaciones para la organización de la información.
Actividad 2 Crear las tarjetas.	Se crean las tarjetas. Se valida que no exista más de una tarjeta con el mismo nombre en cada ejercicio.
Actividad 3 Crear un EOT con categorías predefinidas	Se crean las categorías y el EOT Se valida que no exista más de una categoría con el mismo nombre en cada ejercicio.
Actividad 4 Preparar las instrucciones y finalizar el ejercicio.	Se preparan las instrucciones del ejercicio: En el caso de un ejercicio de ordenamiento de tarjetas abierto: los participantes pueden crear nuevas categorías. En el caso de un ejercicio de ordenamiento de tarjetas cerrado: los participantes no pueden crear nuevas categorías.

Flujograma Anexo 1

2.3.2 Ejecución EOT

Ficha de Proceso	
Proceso:	Ejecución del ordenamiento de tarjetas
Entradas:	Ejercicio de ordenamiento de tarjetas preparado por el arquitecto de información.
Salidas:	Informe de resultados con las clases generadas por los participantes. Ejercicio de ordenamiento de tarjetas ejecutado por los participantes.
Involucrados:	Arquitecto de la información. Participantes
Descripción	
Actividad 1 Instruir a los participantes	Se le explica a los participantes todos los elementos sobre el EOT.
Actividad 2 El participante ejecuta el ejercicio	El participante ejecuta el ejercicio. En caso de ser un EOT abierto el participante crea categorías. No puede existir más de una categoría con el mismo nombre en cada ejercicio. Los participantes crean las clases agrupando las tarjetas sobre las categorías.
Actividad 2 Elaborar informe	Se elabora un informe con el compendio de las clases generadas por los participantes
Flujograma Anexo 2	

2.3.3 Evaluación EOT

Ficha de Proceso	
Proceso:	Evaluación del ordenamiento de tarjetas

Entradas:	Ejercicio de ordenamiento de tarjetas ejecutado por los participantes
Salidas:	Informe de recomendaciones para la organización de la información de acuerdo con los patrones de agrupamiento de los participantes.
Involucrados:	Arquitecto de la información.
Descripción	
Actividad 1 Recoger resultados de la ejecución.	Se recogen las soluciones del EOT.
Actividad 2 Aplicar algoritmo K-Means	Se aplica el algoritmo de formación de clústeres K – Means.
Actividad 3 Aplicar Algoritmos jerárquicos	Se aplican algoritmos aglomerados jerárquicos.
Actividad 4 Elaborar Informe final	Se elabora un informe final de recomendaciones para la organización de la información.
Flujograma Anexo 3	

El análisis de secuencia funciona muy parecido a la técnica de OT, lo que cambia es el mapa mental del usuario. Ahora los resultados se centrarán en la búsqueda de posiciones para los elementos a evaluar. También cuenta con tres procesos fundamentales:

EAS: ejercicio de análisis de secuencia.

- El proceso de preparación de un ejercicio de análisis de secuencia

2.3.4 Preparación EAS

Ficha de Proceso	
Proceso:	Preparación del análisis de secuencia.
Entradas:	El arquitecto de información prepara el EAS con el nombre y la

	descripción del mismo.
Salidas:	Se obtiene un EAS preparado.
Involucrados:	Arquitecto de la información.
Descripción	
Actividad 1 Realizar solicitud de recomendaciones para la organización de la información.	El arquitecto de la información realiza una solicitud de recomendaciones para la organización de la información.
Actividad 2 Crear etiquetas.	Se crean las etiquetas y se valida que no exista más de una etiqueta con el mismo nombre en cada ejercicio.
Actividad 3 Crear posiciones del EAS	Se crean las posiciones y se valida que no exista más de una posición con el mismo nombre en cada ejercicio.
Actividad 4 Finalizar ejercicio	Se finaliza el EAS.
Flujograma Anexo 4	

2.3.5 Ejecución EAS

Ficha de Proceso	
Proceso:	Ejecución de un EAS.
Entradas:	Ejercicio de AS preparado por el arquitecto de información.
Salidas:	Informe de resultados con las clases generadas por los participantes. Ejercicio de ordenamiento de tarjetas ejecutado por los participantes.
Involucrados:	Arquitecto de la información. Participantes
Descripción	
Actividad 1 Instruir a los participantes	Se inicia cuando los participantes son instruidos sobre el EAS y este le es entregado.

Actividad 2 El participante ejecuta el ejercicio	Los participantes crean las clases agrupando las etiquetas sobre las posiciones.
Actividad 3 Elaborar informe.	Se elabora un informe con el compendio de las clases generadas por los participantes.

Flujograma Anexo 5

2.3.6 Evaluación EAS

Ficha de Proceso	
Proceso:	Evaluación de un EAS.
Entradas:	Ejercicio de AS ejecutado por los participantes.
Salidas:	Informe de recomendaciones para la organización de la información de acuerdo con los patrones de agrupamiento de los participantes.
Involucrados:	Arquitecto de la información.

Descripción	
Actividad 1 Recoger resultados	Se recogen los datos de las soluciones del EAS.
Actividad 2 Aplicar evaluación de	Se realiza la evaluación del ejercicio ejecutado por los participantes.
Actividad 3 Elaborar informe	Se elabora informe con recomendaciones para el diseño.
Flujograma Anexo 6	

2.4 Reglas del negocio:

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

- En el ejercicio de ordenamiento de tarjetas abierto: los participantes pueden crear nuevas categorías.
- En el 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.

En el proceso de análisis de secuencia están presentes las siguientes reglas:

- Una etiqueta 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 etiqueta con el mismo nombre en cada ejercicio.
- No puede existir más de una posición con el mismo nombre en cada ejercicio.
- Los participantes no pueden eliminar las posiciones creadas por el diseñador de ejercicios de ordenamiento de tarjetas.

2.5 Requisitos del *software*

Un requisito es un atributo necesario en un sistema, una instrucción que identifica una capacidad de, típico, o factor de calidad de un sistema con el fin de que tenga valor y la utilidad de un cliente o usuario (Pressman, 2010).

2.5.1 Técnicas para la captura de requisitos:

Las técnicas de captura para requisitos sirven para conseguir las principales características que desea el cliente en el sistema y recopilar el conocimiento sobre los requisitos del mismo.

Algunas técnicas para la captura de requisitos son (Blanco, Ruiz, 2006)***:***

- Entrevistas: Intento sistemático de recoger información de otra persona a través de una comunicación interpersonal que se lleva a cabo por medio de una conversación estructurada con el cliente u otros involucrados.

- Tormenta de ideas: Se realizan en reuniones de grupos de trabajo puede estar o no involucrado el cliente final del producto.
- Prototipado.

Para realizar la captura de requisitos en el ordenamiento de tarjetas y análisis de secuencia se utilizaron tres técnicas principales:

- **La técnica de entrevistas** se utilizó durante la entrevista con el cliente para tener una visión general de las características de la organización y de los procesos que se deseaban automatizar.
- **La técnica de tormenta de ideas** se llevó a cabo en reuniones del equipo de trabajo y futuros usuarios de la aplicación.
- **El estudio de Homólogos** permitió conocer cómo se aplican estas técnicas a nivel mundial y ofreció referencias de diseño.

2.5.2 Requisitos funcionales

"Los requisitos funcionales son los que definen las funciones que el sistema será capaz de realizar, describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Estos requisitos al tiempo que avanza el proyecto de *software* se convierten en los algoritmos, la lógica y gran parte del código del sistema" (Pressman, 2010).

RFHU: requisito funcional de la historia de usuario.

Código	Descripción del requisito funcional	Prioridad: Muy Alta
RFHU1	Crear un nuevo ejercicio de agrupamiento de tarjetas.	
RFHU2	Crear un nuevo ejercicio de análisis de secuencia.	
RFHU3	Mezclar las soluciones de un ejercicio de análisis de secuencia.	
RFHU4	Mostrar las gráficas de evaluación de un ejercicio de agrupamiento de tarjetas según los algoritmos seleccionados.	
RFHU5	Evaluar soluciones de un ejercicio de análisis de secuencia.	

RFHU6	Mostrar lista de soluciones de un ejercicio de agrupamiento de tarjetas.	
RFHU7	Mezclar las soluciones de un ejercicio de agrupamiento de tarjetas.	
Código	**Descripción del requisito funcional**	**Prioridad: Alta**
RFHU8	"Guardar como" para un ejercicio de análisis de secuencia y agrupamiento de tarjetas.	
RFHU9	Terminar la preparación de un ejercicio de agrupamiento de tarjetas abierto.	
RFHU10	Ejecutar ejercicio de agrupamiento de tarjetas cerrado.	
RFHU11	Terminar la ejecución de un ejercicio agrupamiento de tarjetas cerrado.	
RFHU12	Crear una categoría en un ejercicio de agrupamiento de tarjetas abierto en la ejecución.	
RFHU13	Editar una categoría en un ejercicio de agrupamiento de tarjetas abierto en la ejecución.	
RFHU14	Evaluar soluciones de ejercicio de agrupamiento de tarjetas.	
RFHU15	Cargar soluciones a evaluar del ejercicio de agrupamiento de tarjetas que está siendo ejecutado.	
RFHU16	Cargar soluciones a evaluar desde un directorio del disco del ejercicio de agrupamiento de tarjetas que está siendo ejecutado.	
RFHU17	Cargar soluciones a evaluar desde un directorio del disco del ejercicio de análisis de secuencia que está siendo ejecutado.	
RFHU18	Cargar soluciones a evaluar del ejercicio de análisis de secuencia que está siendo ejecutado.	
Código	**Descripción del requisito funcional**	**Prioridad: Media**
RFHU19	Eliminar una categoría en un ejercicio de agrupamiento de tarjetas abierto en la ejecución.	
RFHU20	Eliminar todas las categorías en un ejercicio de agrupamiento de tarjetas abierto en la ejecución.	
RFHU21	Ejecutar un ejercicio de agrupamiento de tarjetas abierto.	
RFHU22	Terminar la ejecución de un ejercicio de agrupamiento de tarjetas abierto.	
RFHU23	Seleccionar una solución de la lista de soluciones de un ejercicio de agrupamiento de	

	tarjetas.
RFHU24	Eliminar una solución de la lista de soluciones de un ejercicio de agrupamiento de tarjetas.
RFHU25	Eliminar fichero de la lista de soluciones a evaluar de un ejercicio de agrupamiento de tarjetas.
RFHU26	Seleccionar el algoritmo *K-Means* para evaluar un ejercicio de agrupamiento de tarjetas.
RFHU27	Insertar datos para evaluar por el algoritmo *K-Means*.
RFHU28	Seleccionar el algoritmo *single link* para evaluar un ejercicio de agrupamiento de tarjetas.
RFHU29	Seleccionar el algoritmo *complete link* para evaluar un ejercicio de agrupamiento de tarjetas.
RFHU30	Eliminar fichero de la lista de soluciones a evaluar de un ejercicio de análisis de secuencia.
RFHU31	Abrir un ejercicio de análisis de secuencia o de agrupamiento de tarjetas.
RFHU32	Seleccionar el algoritmo *group average* para evaluar un ejercicio de agrupamiento de tarjetas.
RFHU33	Preparar ejercicio de análisis de secuencia.
RFHU34	Crear las posiciones en correspondencia con el número de etiquetas y de forma consecutiva en un ejercicio de análisis de secuencia.
RFHU35	Terminar ejecución de ejercicio de análisis de secuencia.
RFHU36	Mostrar lista de soluciones de ejercicio de análisis de secuencia.
RFHU37	Ejecutar ejercicio de análisis de secuencia.
RFHU38	Terminar preparación de un ejercicio de análisis de secuencia.
Código	**Descripción del requisito funcional** **Prioridad: Baja**
RFHU39	Crear etiqueta en un ejercicio de análisis de secuencia.
RFHU40	Editar etiqueta en un ejercicio de análisis de secuencia.
RFHU41	Eliminar una etiqueta en un ejercicio de análisis de secuencia.

RFHU42	Eliminar todas las etiquetas en un ejercicio de análisis de secuencia.
RFHU43	Seleccionar una solución de un ejercicio de análisis de secuencia.
RFHU44	Eliminar una solución de un ejercicio de análisis de secuencia.
RFHU45	Cerrar lista de soluciones de un ejercicio de análisis de secuencia
RFHU46	Salir desde un ejercicio de análisis de secuencia o de agrupamiento de tarjetas.
RFHU47	Crear una tarjeta de un ejercicio de agrupamiento de tarjetas cerrado.
RFHU48	Editar una tarjeta de un ejercicio de agrupamiento de tarjetas cerrado.
RFHU49	Eliminar una tarjeta de un ejercicio de agrupamiento de tarjetas cerrado.
RFHU50	Eliminar todas las tarjetas de un ejercicio de agrupamiento de tarjetas cerrado.
RFHU51	Crear una categoría de un ejercicio de agrupamiento de tarjetas cerrado.
RFHU52	Editar una categoría de un ejercicio de agrupamiento de tarjetas cerrado.
RFHU53	Eliminar una categoría de un ejercicio de agrupamiento de tarjetas cerrado
RFHU54	Eliminar todas las categorías de un ejercicio de agrupamiento de tarjeta cerrado.
RFHU55	Terminar la preparación ejercicio de agrupamiento de tarjetas cerrado.
RFHU56	Crear una tarjeta de un ejercicio de agrupamiento de tarjetas abierto.
RFHU57	Editar una tarjeta de un ejercicio de agrupamiento de tarjetas abierto.
RFHU58	Eliminar una tarjeta de un ejercicio de agrupamiento de tarjetas abierto.
RFHU59	Eliminar todas las tarjetas de un ejercicio de agrupamiento de tarjetas abierto.
RNF (Requisitos no funcionales)	
	Usabilidad
	El sistema puede ser utilizado por cualquier persona que posea conocimientos básicos para interactuar con una computadora.

RNF1	El diseño debe ser sencillo sin mucha inversión de tiempo de entrenamiento.
	Fiabilidad
RNF2	El sistema debe estar 100 % disponible, las 24 horas de un día, siempre que esté instalado en la estación de trabajo, tanto para los Arquitectos de Información como a los miembros de los participantes para ser utilizado.
	Soporte
RNF3	El sistema debe ejecutarse en un ordenador que tenga 100 MB libres en el disco duro como mínimo.
RNF4	El sistema debe ejecutarse en un ordenador que tenga 256 MB de RAM como mínimo.
	Restricciones de diseño
RNF5	El sistema se desarrolla en el lenguaje Java, sobre la plataforma de desarrollo J2SE 1.4 o superior.
	Interfaz
RNF6	El sistema debe contar con las interfaces de usuarios diseñados de la manera más atractiva posible.
RNF7	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.
RNF8	Para texto se utiliza el tipo de fuente arial con tamaño mínimo 11 y máximo 14.
RNF9	El color de los textos variará de acuerdo con el color de fondo.
RNF10	Los íconos de las tarjetas, etiquetas, posiciones y categorías deben ser comprensibles, de manera que especifiquen la funcionalidad que encierran cada uno de estos.
RNF11	Debe predominar el uso de los íconos para la interacción con el usuario.

Tabla 2. LRP: Lista de reserva del producto.

2.6 Historias de usuarios (HU)

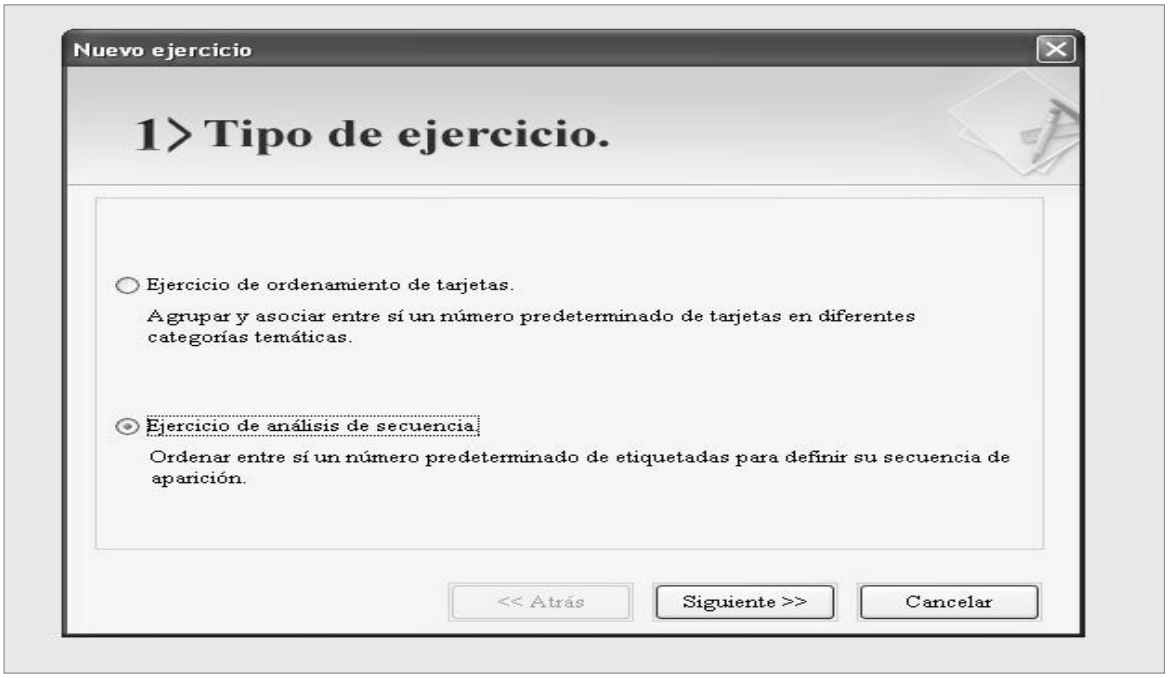
Las historias de usuarios son utilizadas como el único documento de requisitos que se genera en SXP. Son escritas en lenguaje natural, no excediendo su tamaño de unas pocas líneas de texto. Las historias de usuario guían la construcción de las pruebas de aceptación, elemento clave en XP (deben generarse una o más *pruebas* para verificar que la historia de usuario ha sido correctamente implementada) y son utilizadas para estimar tiempos de desarrollo. En este sentido, proveen detalles

suficientes para hacer una estimación razonable del tiempo que llevará implementarla. En el momento de implementar se deben detallar a través de la comunicación con el cliente. Son la base para las pruebas funcionales (Peñalver, G., Meneses, A., García, S. 2010).

A continuación se muestran algunas de las HU más importantes del sistema propuesto para un mejor entendimiento del mismo.

Historia de Usuario	
Código: HU1	**Nombre Historia de Usuario:** Crear un nuevo ejercicio
Modificación de Historia de Usuario número: Ninguna	
Referencia: RFDC1, RFDC2	
Programador: Frank Balmaseda Borlado	**Iteración asignada:** 1
Prioridad: Alta	**Puntos estimados:** 1
Riesgo en desarrollo: Baja	**Puntos reales:**
Descripción: 1-Desde el **menú principal** selecciona opción **nuevo** para crear un ejercicio. 2-Se muestra una pantalla que permite **escoger el tipo de ejercicio** que puede ser de agrupamiento de tarjetas o análisis de secuencia, se selecciona el **botón siguiente.** 3- Se muestra una pantalla para poner un nombre y las orientaciones del mismo. 3.1- En el caso de agrupamiento de tarjetas **debe marcar si es de tipo abierto o cerrado**, se selecciona el **botón siguiente** para continuar. 4- En la nueva pantalla **se especifica el directorio en el disco donde salvará el fichero** en formato xml. 5- Se selecciona el **botón finalizar** terminando el proceso de creación del ejercicio. Luego se muestra una pantalla con los datos del ejercicio creado.	
Observaciones: En el agrupamiento de tarjetas debe seleccionar el tipo de ejercicio abierto o cerrado. El campo del nombre del ejercicio no puede estar vacío, los caracteres pueden ser alfanuméricos. Debe seleccionar el directorio en disco para guardar el fichero.	

Prototipo de interfaz de usuario:



Ver Anexo 7,8,9,10

Historia de Usuario	
Código: HU2	**Nombre Historia de Usuario:** Crear una tarjeta
Modificación de Historia de Usuario número: Ninguna	
Referencia: RFDC1, RFDC47, RFDC56	
Programador: Frank Balmaseda Borlado	**Iteración asignada:** 1
Prioridad: Alta	**Puntos estimados:** 1
Riesgo en desarrollo: Baja	**Puntos reales:**

Descripción:

1- Para crear la tarjeta se da **clic sobre el ícono** que representa la **adición de una tarjeta** o con **clic derecho** sobre el área de trabajo se puede seleccionar la opción crear nueva tarjeta.

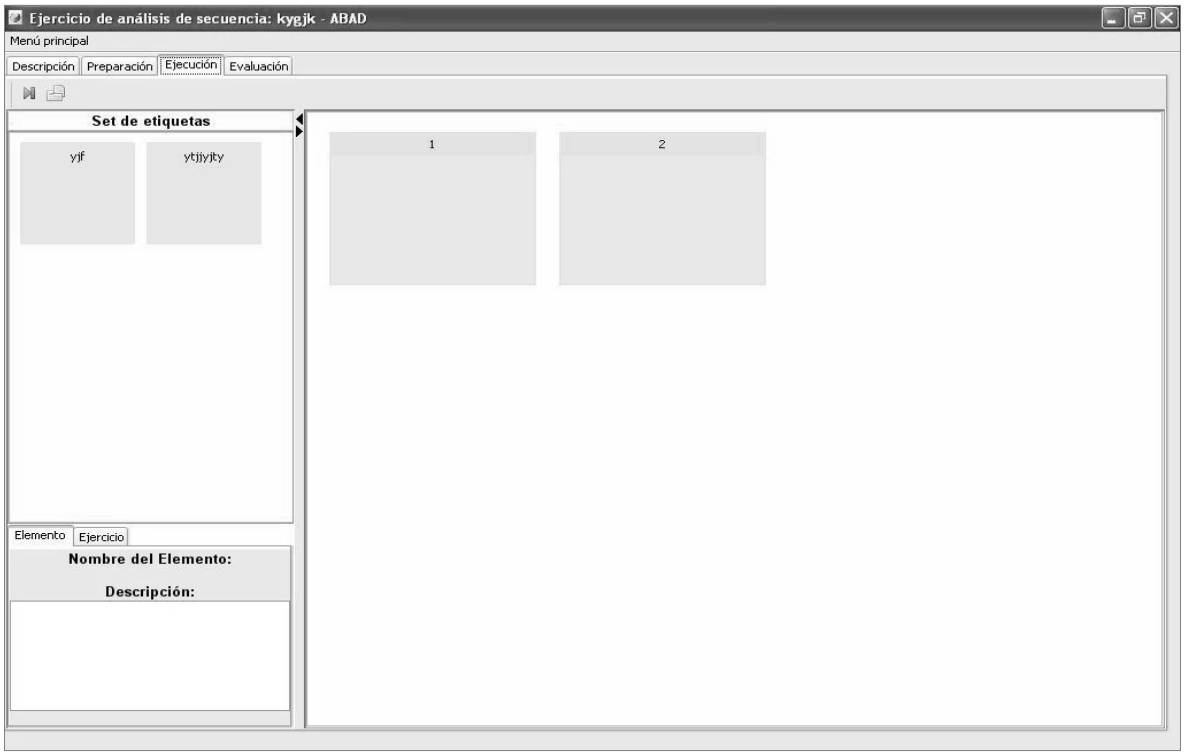
2- Se muestra un cuadro de diálogo con dos campos uno para el nombre y otro para la descripción de lo que representa la tarjeta, la descripción no es un campo obligatorio y para rellenar este campo se debe seleccionar la opción **incluir descripción.**

3- El nombre de la tarjeta no puede repetirse en el ejercicio y no debe quedar en blanco, si esto ocurre en ambos casos se muestra un mensaje de error en la misma ventana donde se especifica el

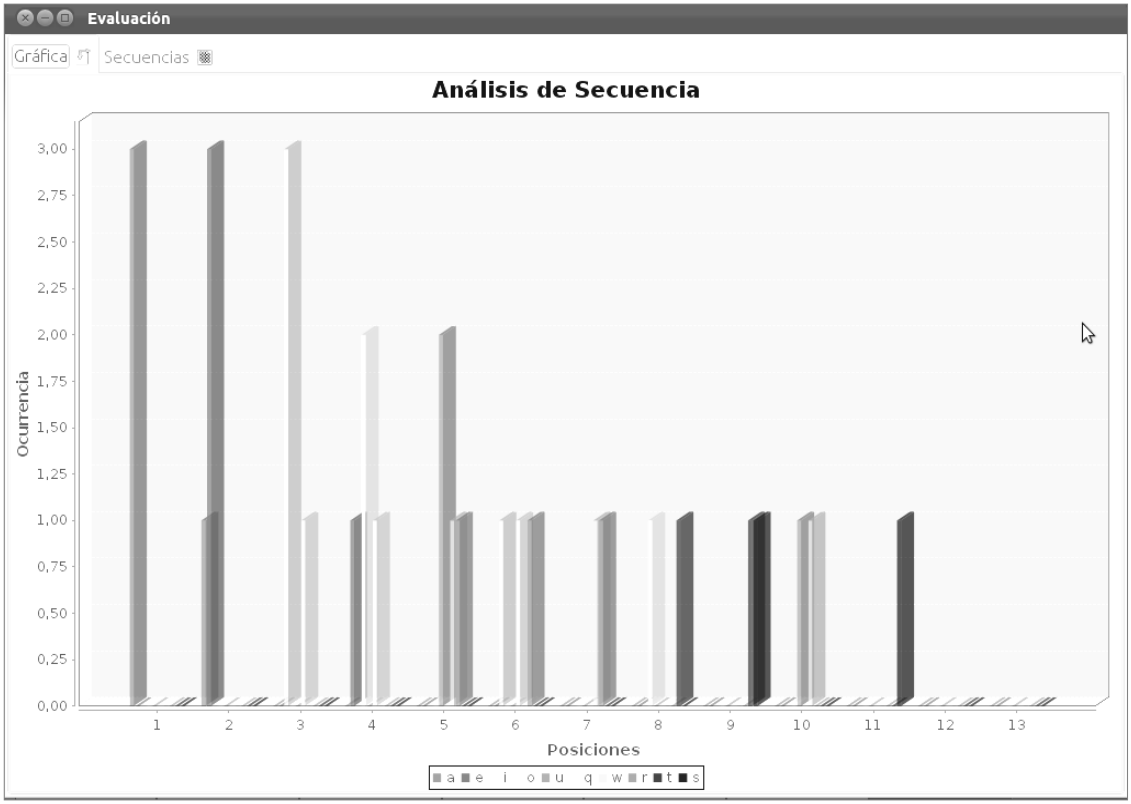
error en que está incurriendo. 4- Completados todos los campos se procede a seleccionar el **botón aceptar** quedando creada la tarjeta, en caso de no querer terminar el proceso de creación de dicha tarjeta se selecciona el **botón cancelar**.
Observaciones: Para crear una tarjeta debe estar creado el ejercicio. Las tarjetas no pueden estar repetidas. El nombre de la tarjeta es un campo obligatorio.
Prototipo de interfaz de usuario: Anexo 11

Historia de Usuario	
Código: HU26	**Nombre Historia de Usuario:** Ejecutar ejercicio de análisis de secuencia
Modificación de Historia de Usuario número: Ninguna	
Referencia: RFDC35, RFDC37, RFDC45, RFDC46	
Programador: Frank Balmaseda Borlado	**Iteración asignada:** 1
Prioridad: Alta	**Puntos estimados:** 1
Riesgo en desarrollo: Alta	**Puntos reales:**
Descripción: 1- Arrastrar una etiqueta del *set* de etiquetas a una posición	
Observaciones: Debe estar terminada la preparación del ejercicio	

Prototipo de interfaz de usuario:



	Historia de Usuario
Código: HU32	**Nombre Historia de Usuario:** Evaluar ejercicio de análisis de secuencia.
Modificación de Historia de Usuario número: Ninguna	
Referencia: RFDC5, RFDC17, RFDC18, RFDC30, RFDC36, RFDC43, RFDC44, RFDC45	
Programador: Frank Balmaseda Borlado	**Iteración asignada:** 1
Prioridad: Alta	**Puntos estimados:** 1
Riesgo en desarrollo: Alta	**Puntos reales:**
Descripción: 1- Se selecciona el ícono de evaluar	

2- Se muestra la ventana con la gráfica de evaluación y la tabla de ocurrencia de las soluciones
Observaciones: Deben existir soluciones en la lista para que se active el ícono de evaluación
Prototipo de interfaz de usuario: 

	Historia de Usuario
Código: HU19	**Nombre Historia de Usuario:** Evaluar ejercicio de agrupamiento de tarjetas
Modificación de Historia de Usuario número: Ninguna	
Referencia: RFDC4, RFDC6, RFDC7, RFDC14, RFDC15, RFDC16, RFDC23, RFDC24, RFDC25, RFDC26, RFDC27, RFDC28, RFDC29, RFDC32	
Programador: Frank Balmaseda Borlado	**Iteración asignada:** 1

Prioridad: Alta	**Puntos estimados:** 1
Riesgo en desarrollo: Alta	**Puntos reales:**
Descripción: 1- Se selecciona al menos un algoritmo de evaluación o todos	
Observaciones: Deben existir soluciones en la lista para que se active el ícono de evaluación	
Prototipo de interfaz de usuario: Anexo 12	

2.7 Conclusiones del capítulo

Con la realización de este capítulo se concluye que:

- La descripción de la solución propuesta brinda un mejor entendimiento a los desarrolladores de las expectativas del cliente.
- La descripción de los procesos del negocio permite un mejor entendimiento de las actividades a automatizar.
- La descripción de las historias de usuarios guía a los clientes y desarrolladores por la aplicación.
- La obtención de los requisitos funcionales y no funcionales del sistema permite la implementación paralela de la herramienta.

Capítulo 3 Diseño del sistema

3.1 Introducción

En este capítulo se presentan y fundamentan los patrones de diseño utilizados, la validación de los métodos matemáticos de agrupamientos jerárquicos propuestos para el proceso de evaluación de los ejercicios de OT descritos en el capítulo 1. Así como una descripción de la propuesta de la arquitectura y paquetes de diseño de la solución a desarrollar, los diseños de casos de pruebas basados en las historias de usuarios y la validación de los requisitos.

3.2 Validación de requisitos

La validación de los requisitos es muy importante para la mejor implementación del producto y asegurar la calidad del mismo ya que una vez validados se puede concluir que el sistema cumplirá con las expectativas del cliente.

“Ningún método de revisión es universalmente correcto, ya que los proyectos difieren en su flexibilidad de funciones, personal, presupuesto, cronograma, y la calidad” (Wiegers, 2003).

Existen diversas técnicas de validación de requisitos como son las revisiones, el prototipado, la auditoría a los requisitos, entre otras. Atendiendo a las características de la solución propuesta y el entorno en el que se desarrolla se utilizaron las siguientes técnicas para validar los requisitos capturados en el capítulo anterior.

- **Revisiones:** el equipo del proyecto en conjunto con el cliente realizó la revisión de los requisitos capturados avalando la buena redacción y descripción de los mismos.
- **Prototipado:** la realización de los prototipos permitió corregir los errores de comprensión y redacción de los requisitos capturados hasta el momento, así como añadir los que no se habían definido, estos fueron presentados al cliente quien señaló los errores detectados y avaló el prototipo presentado.

3.3 Validación de los métodos matemáticos utilizados en el proceso de evaluación de un ejercicio de ordenamiento de tarjetas

Para la validación de los métodos matemáticos utilizados en la evaluación en la herramienta K-sort, se compararon los datos obtenidos con la herramienta matemática R[17]. Se introdujeron los mismos datos de la matriz de ocurrencia[18] y se le realizó todo el proceso de evaluación. Se tomó una muestra de 35 soluciones y se analizó el ejercicio en la solución propuesta y en el R obteniéndose los mismos dendogramas en las herramientas por cada método.

A continuación se muestra el caso de estudio realizado:

Tarjetas:

Motor

Bujías

Volante

Salpicadero

A partir de los agrupamientos de los usuarios se conforma la matriz de ocurrencia que representa el número de veces que las tarjetas aparecieron en el mismo grupo, por ejemplo la tarjeta **motor** y **bujías** se agruparon juntas 17 veces.

Matriz de ocurrencia

	Bujías	**Motor**	**Volante**	**Salpicadero**
Bujías	0	17	0	0
Motor	17	0	4	2
Volante	0	4	0	12
Salpicadero	0	2	12	0

Para el caso de la evaluación mediante el algoritmo *Single Link* los puntos de agrupamiento obtenidos en los dendogramas de la herramienta R (figura 1) coinciden con los puntos de agrupamiento obtenidos en los dendogramas de la aplicación K-sort (figura 2).

[17] Conjunto de módulos estadísticos que, mediante las interfaces de que dispone, permite realizar análisis de datos y representación de los mismos.

[18] Aparición conjunta de las tarjetas.

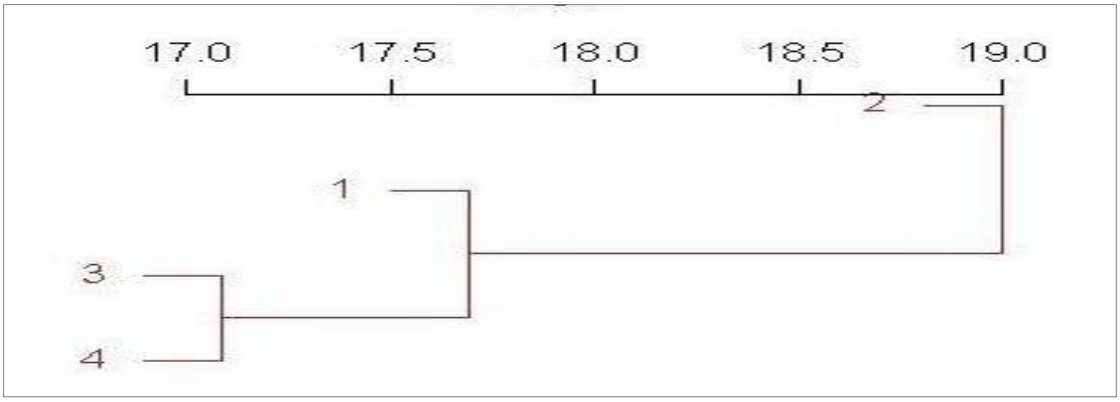


Figura. 1 Dendograma obtenido con el algoritmo *Single Link* en la herramienta R

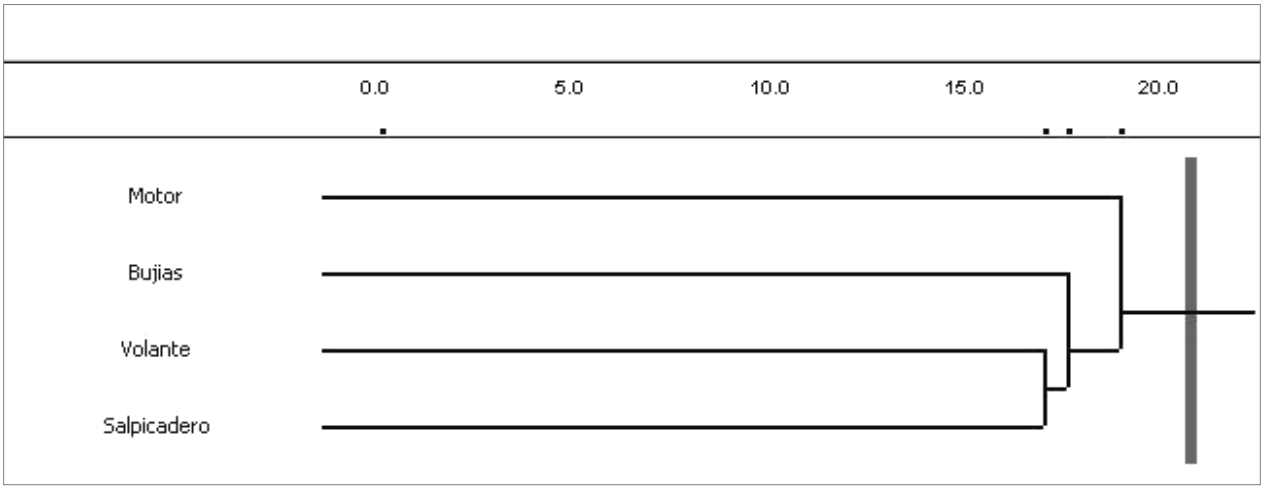


Figura. 2 Dendograma obtenido con el algoritmo *Single Link* en la herramienta K-sort

Para el caso de la evaluación mediante el algoritmo *Complete Link* los puntos de agrupamiento obtenidos en los dendogramas en la herramienta R (figura 3) coinciden con los puntos de agrupamiento obtenidos en los dendogramas en la aplicación K-sort (figura 4).

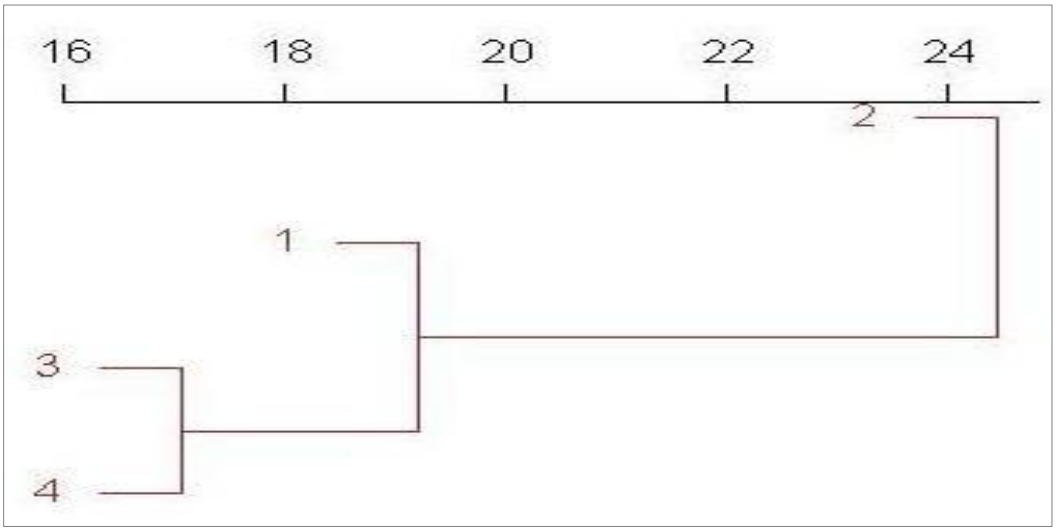


Figura. 3 Dendograma obtenido con el algoritmo *Complete Link* en la herramienta R

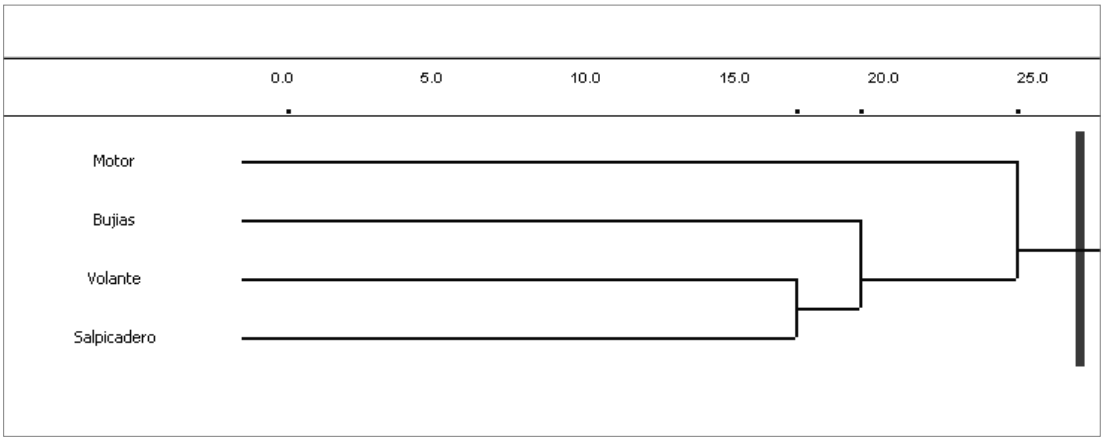


Figura. 4 Dendograma obtenido con el algoritmo *Complete Link* en la herramienta K-sort

Para el caso de la evaluación mediante el algoritmo *Group average*, los puntos de agrupamiento obtenidos en los dendogramas en la herramienta R (figura 5) coinciden con los puntos de agrupamiento obtenidos en los dendogramas de la aplicación K-sort (figura 6).

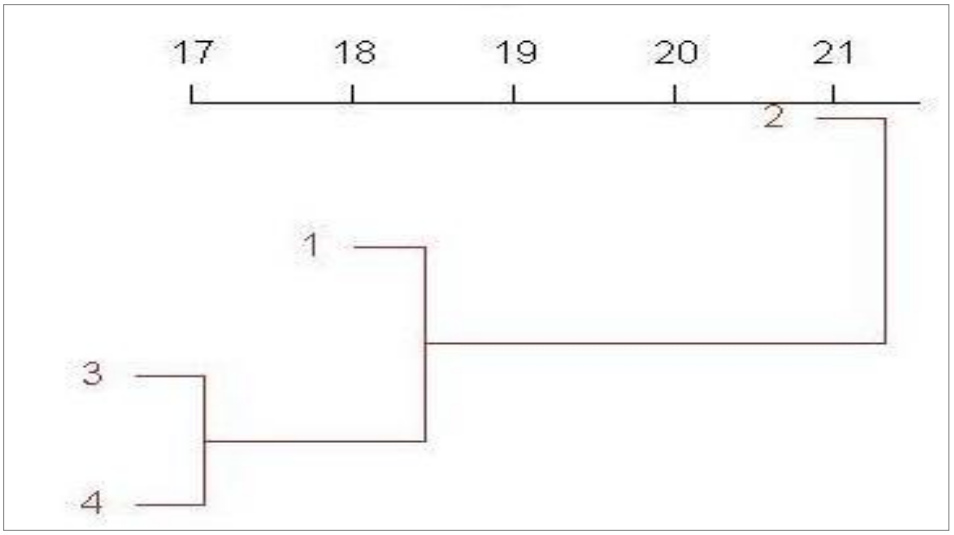


Figura. 5 Dendograma obtenido con el algoritmo *Group average* en la herramienta R

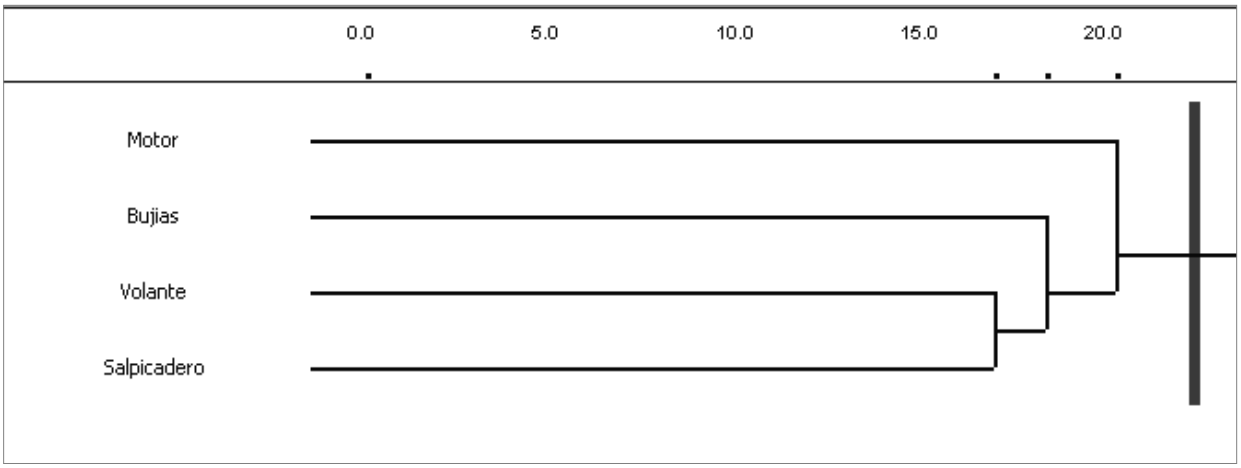


Figura. 6 Dendograma obtenido con el algoritmo *Group average* en la herramienta K-sort

Como se puede observar los resultados de los agrupamientos de los algoritmos propuestos para la evaluación de la técnica de ordenamiento de tarjetas en la aplicación K-sort coinciden en su totalidad con los resultados obtenidos en la herramienta R, lo que valida el buen funcionamiento de los mismos y su utilización para la obtención de los agrupamientos durante el proceso de evaluación de un ejercicio de ordenamiento de tarjetas.

3.4 Patrones de diseño

3.4.1 Patrones GRASP[19]

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos (Larman, 1999). En el desarrollo de esta solución se propone la aplicación de los patrones GRASP: Experto, Creador, Controlador, Bajo Acoplamiento, Alta Cohesión, Indirección y Polimorfismo.

El patrón Experto resuelve el problema de asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se conecte con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento (Larman, 1999).

El patrón Bajo Acoplamiento recomienda asignar las responsabilidades de tal forma que se le dé soporte a una mayor reutilización y poca dependencia. Con el patrón Alta Cohesión se mejoran la claridad y facilidad con que se entiende el diseño, se simplifica el mantenimiento y las mejoras de funcionalidad. A menudo, se genera un bajo acoplamiento que soporta mayor capacidad de 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 Polimorfismo recomienda la definición de un método homónimo en la clase generalización y en cada subclase con un comportamiento específico (Larman, 1999).

3.4.2 Patrones GoF[20]

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).

3.5 Patrones de arquitectura

Existen 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

[19] En español, patrones generales de *software* para la asignación de responsabilidades
[20] *Gang of Four,* en español “La pandilla de los cuatro”

ante problemas comunes de organización y flujo de datos entre componentes de un sistema informático. Son comúnmente llamados patrones de arquitectura.

3.5.2 Patrón arquitectónico de Tres Capas

Se basa en la división en el nivel de acceso a datos, nivel de lógica de negocio y nivel de presentación lo cual permite que sean creados sistemas con un bajo acoplamiento entre sus módulos o componentes.

“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).

Ventajas que presenta la arquitectura en tres capas:

- Independencia de las capas: La separación de roles en tres capas, hace más fácil reemplazar o modificar una capa sin afectar a las capas restantes.
- Se pueda descomponer la aplicación en varios niveles de abstracción.
- Facilita la evolución del sistema, ya que los cambios solo deben de afectar a la capa donde se encuentre la modificación.

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. Fue definida por el proyecto.

3.6 Modelación del sistema propuesto

El sistema propuesto se divide en tres módulos: preparación, ejecución y evaluación de ejercicios de ordenamiento de tarjetas y análisis de secuencia. El módulo de preparación de un ejercicio de ordenamiento de tarjetas y análisis de secuencia se encarga de todo lo relacionado con el diseño de los ejercicios. Con este módulo se pueden elaborar las tarjetas, etiquetas, posiciones y las categorías necesarias para ejecutar un ejercicio de OT y AS, así como, definir el tipo de ejercicio de ordenamiento de tarjetas, abierto o cerrado.

El módulo de ejecución de EOT y EAS permite que los participantes agrupen las tarjetas en clases haciendo uso de las categorías para el primer caso y que posicionen las etiquetas haciendo uso de las posiciones para el segundo. 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 EOT y el EAS resuelto por cada participante.

Mediante el módulo de evaluación de los ejercicios de ordenamiento de tarjetas y de análisis de secuencia 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 o análisis de secuencia, para su posterior análisis. Con este módulo se obtienen los clústeres luego de aplicar algoritmos de aglomerados jerárquicos y K – Means a dichos agrupamientos.

3.7 Propuesta de la arquitectura

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. 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 como se observa en la (figura 7).

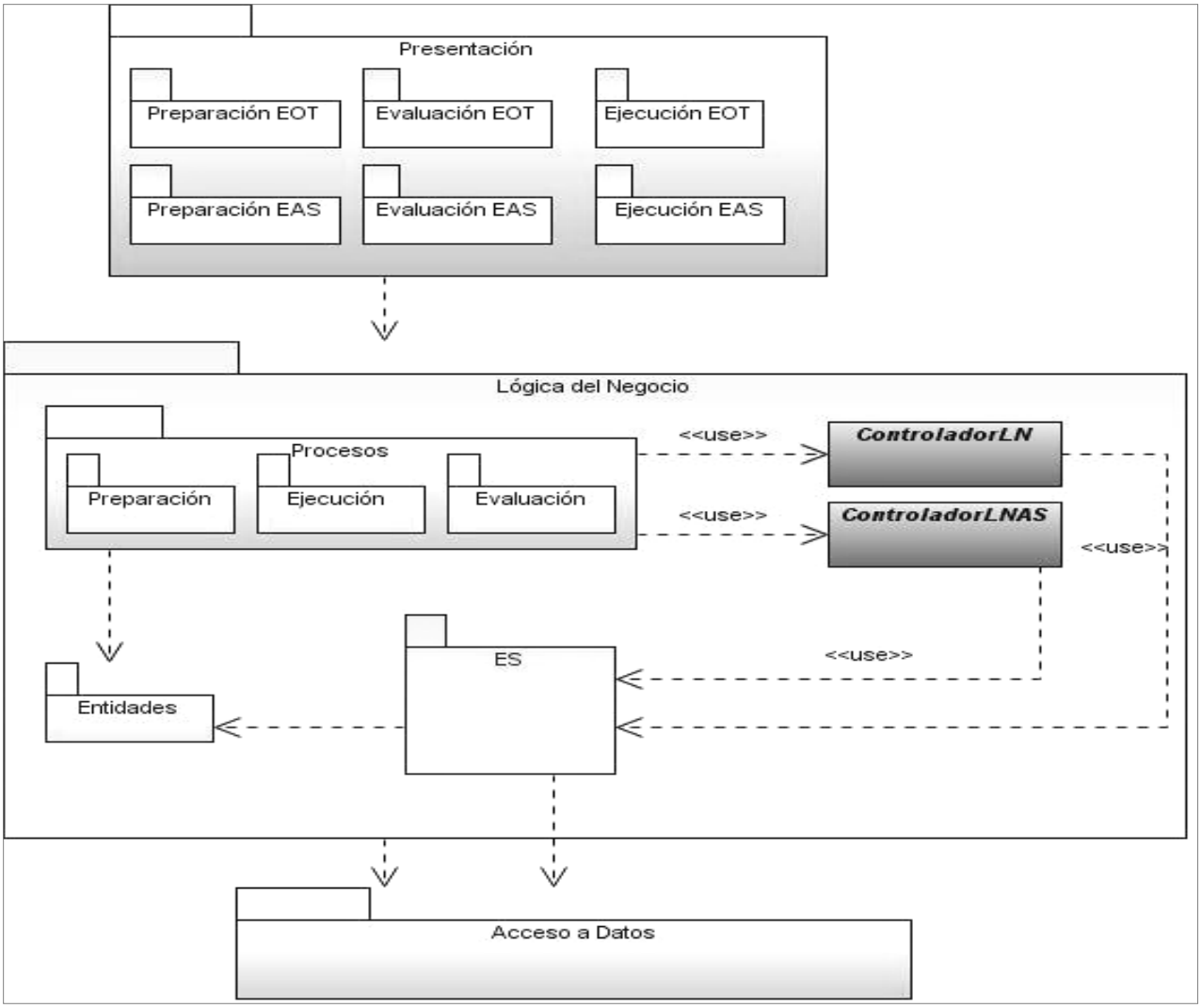


Figura. 7 Modelo de diseño.

A continuación se describen los paquetes con que contará la aplicación:

- **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, PreparaciónAS, EjecuciónAS, EvaluaciónAS y el paquete Elementos Genéricos (figura 8).

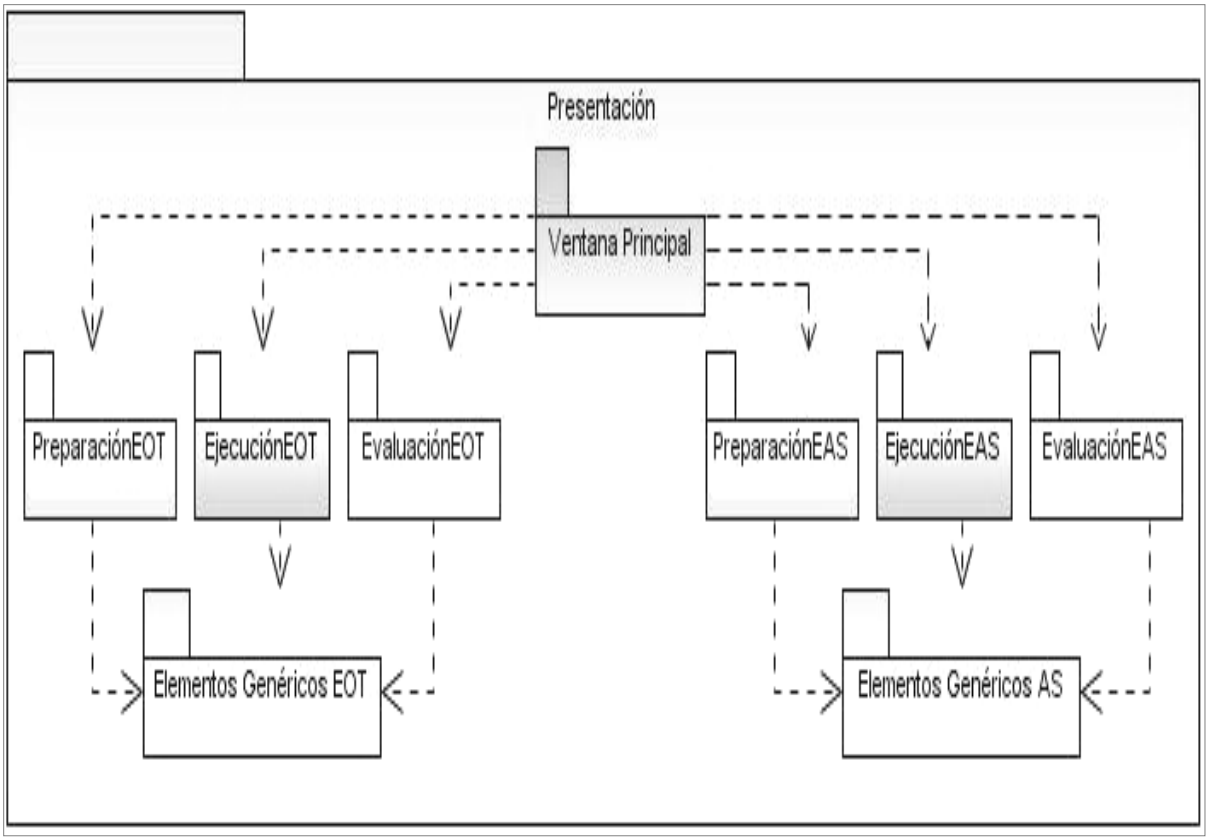


Figura. 8 Capa presentación paquete preparación

- **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 (figura 9) (figura 10) (figura 11), Entidades (figura 12) y Entrada Salida (ES) (figura 13).

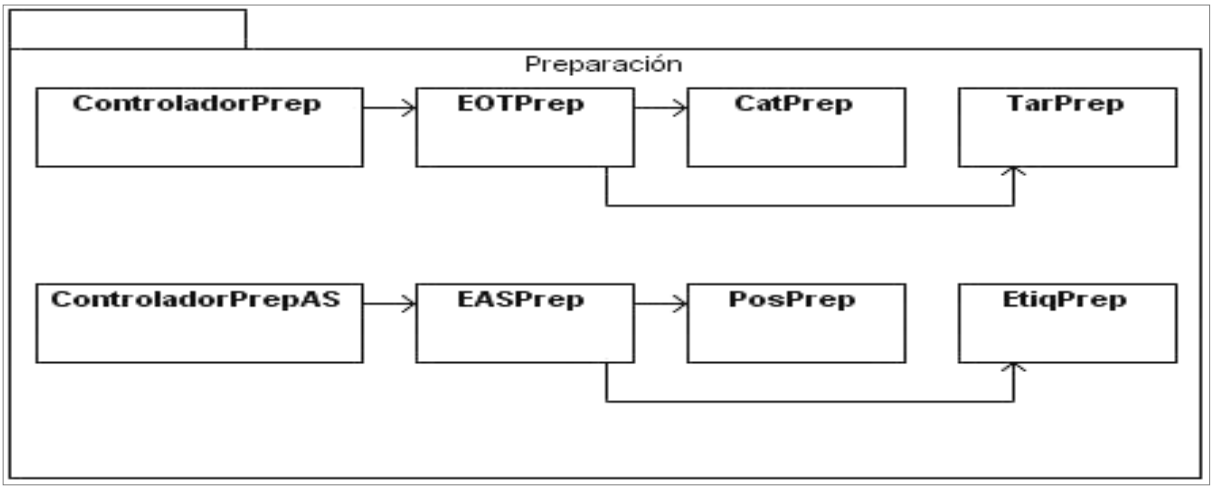


Figura. 9 Capa lógica del negocio proceso preparación

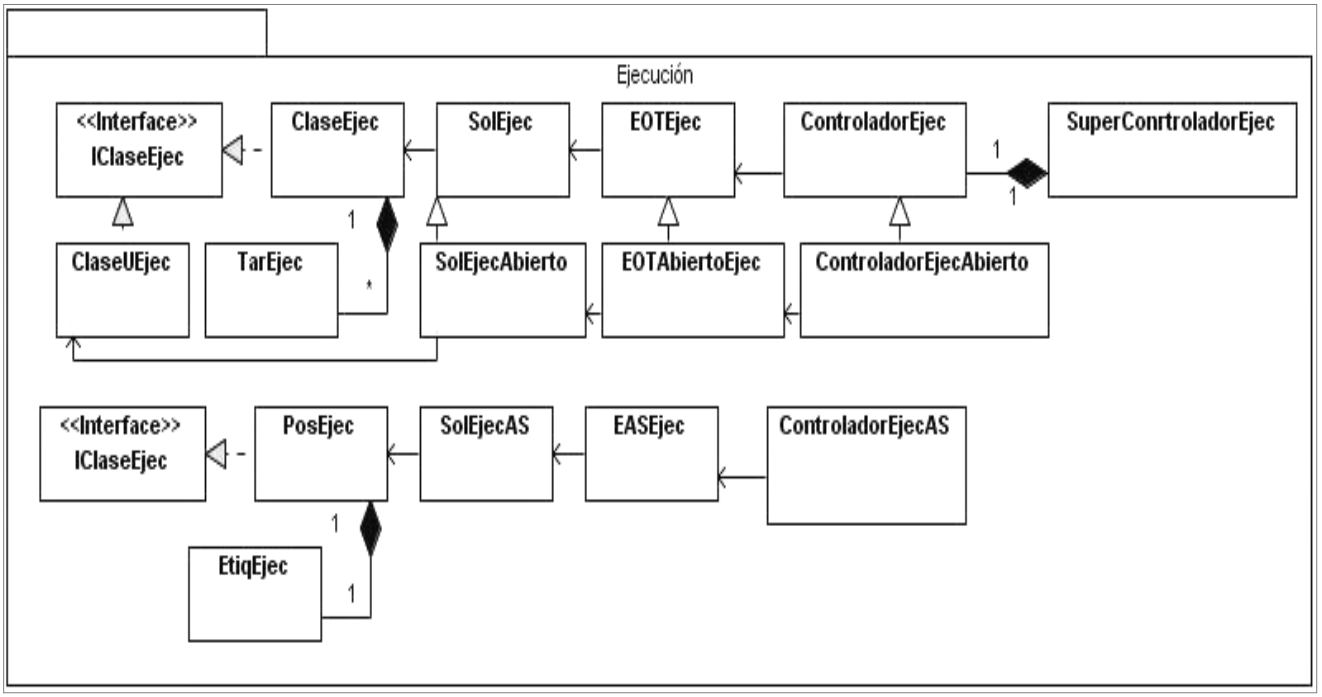


Figura. 10 Capa lógica del negocio proceso ejecución

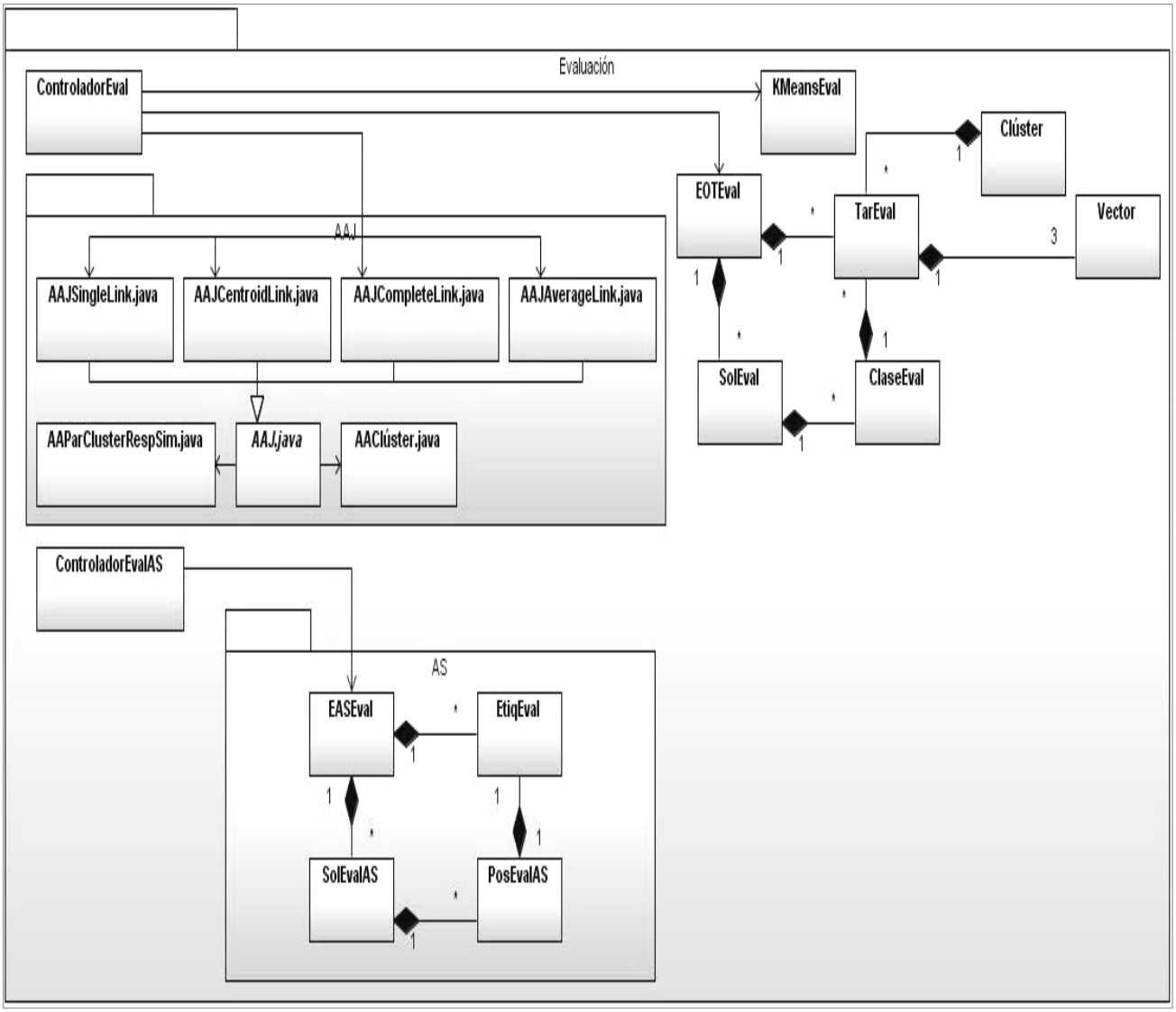


Figura. 11 Capa lógica del negocio proceso evaluación

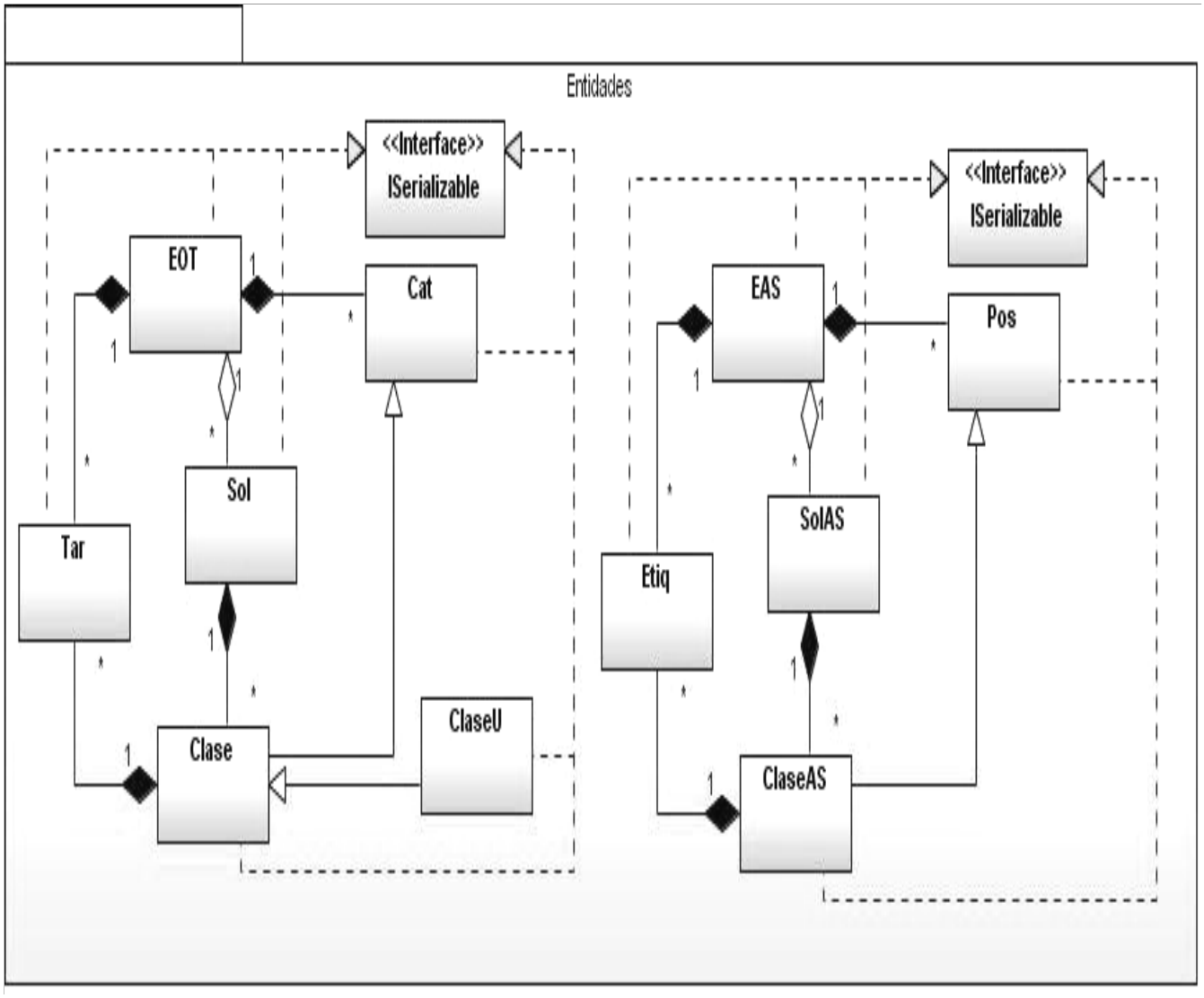


Figura. 12 Capa lógica del negocio. Entidades

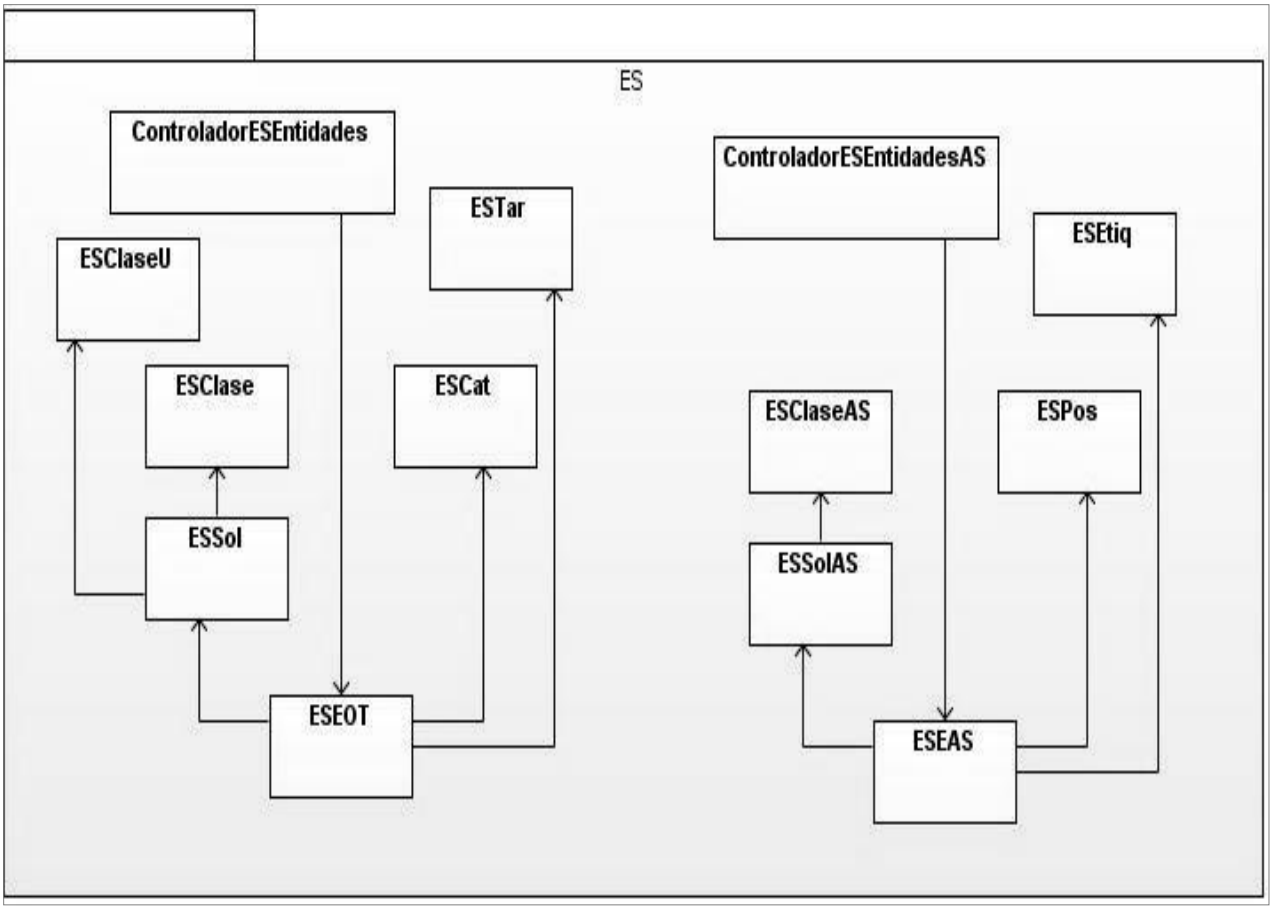


Figura. 13 Capa lógica del negocio. Entrada y salida

- **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 y del análisis de secuencia. Estos ficheros pueden contener EOT y EAS 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 como se muestra en la (figura 14).

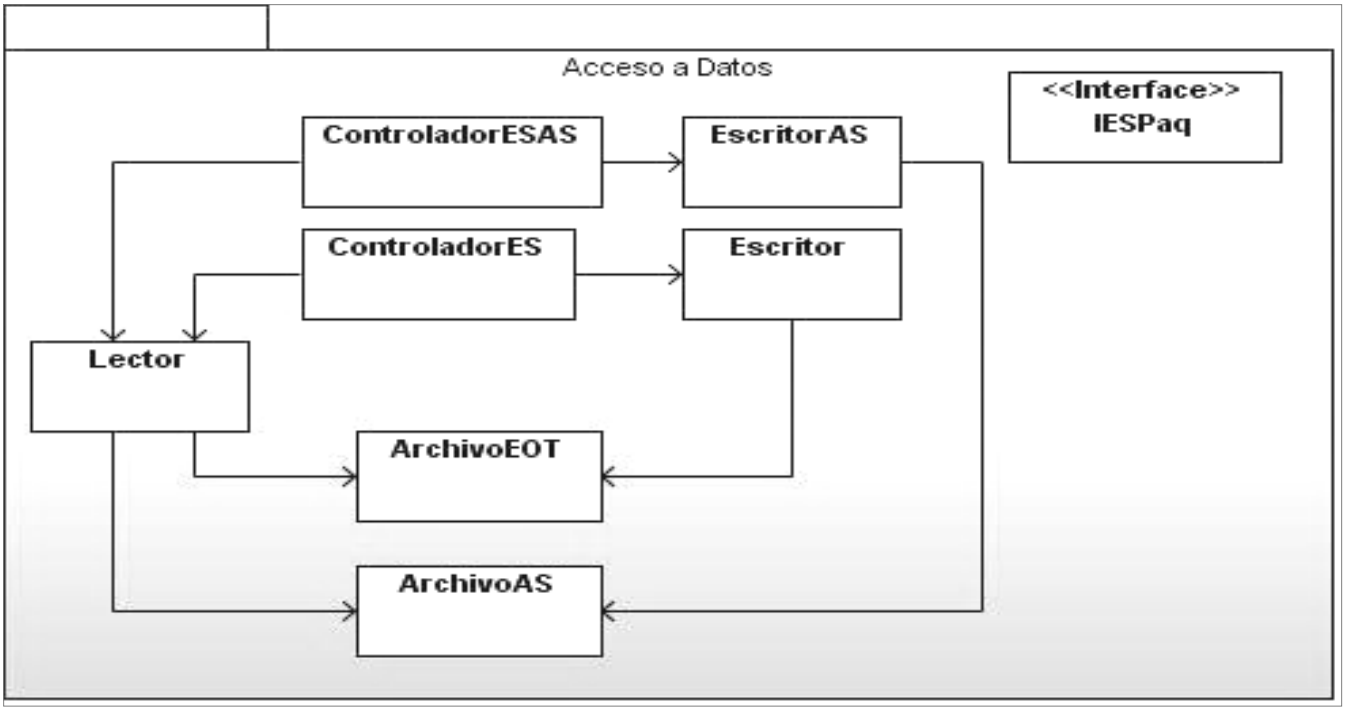


Figura. 14 Capa de acceso a datos

3.8 Pruebas

3.8.1 Pruebas de aceptación

Las pruebas de aceptación constituyen pruebas de caja negra que se centran en el cumplimiento de los requisitos definidos para el sistema una vez concluido. Por lo tanto, estas pruebas constituyen la validación de la aplicación por parte del usuario final y según Pressman plantean lo siguiente:

"La validación del *software* se consigue mediante una serie de pruebas de caja negra que demuestran la conformidad con los requisitos. Un plan de prueba traza la clase de pruebas que se han de llevar a cabo y un procedimiento de prueba define los casos de prueba específicos en un intento por descubrir errores de acuerdo con los requisitos".

Tanto el plan como el procedimiento estarán diseñados para asegurar que se satisfacen todos los requisitos funcionales, que se alcanzan todos los requisitos de rendimiento, que la documentación es correcta e inteligible y que se alcanzan otros requisitos (por ejemplo, portabilidad, compatibilidad, recuperación de errores, facilidad de mantenimiento)" (Pressman, 1998).

En SXP estas pruebas se realizan entre iteraciones y son las que definen el paso a la próxima iteración. Estas pruebas pueden aplicarse durante semanas o meses y van desde un informal "paso de prueba" hasta la ejecución continua de una serie de pruebas bien planificadas (Peñalver, G., Meneses,

A., García, S. 2010). A continuación se muestran algunos de los diseños de casos de pruebas realizados, basados en algunas de las historias de usuarios descritas en el capítulo anterior.

3.8.2 Diseño de caso de prueba crear ejercicio

Flujo central

1- La aplicación muestra la ventana principal con el menú principal.
2- El usuario selecciona la opción nuevo ejercicio
3- El usuario selecciona el tipo de ejercicio.

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario llena las opciones de crear ejercicio, dando luego en el botón aceptar.	Se crea el ejercicio correctamente.	*(Satisfactoria/No satisfactoria)*	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario deja el campo nombre: “vacío” dando luego en el botón aceptar.	El sistema muestra el mensaje “Debe introducir un nombre”.	*Satisfactoria/No satisfactoria)*	
El usuario crea ejercicio con el mismo nombre, dando luego en el botón aceptar.	El sistema muestra el mensaje El ejercicio que desea guardar ya existe ¿Desea sobre escribirlo?¨	*Satisfactoria/No satisfactoria)*	
Evaluación de la Prueba:	*(Satisfactoria/No satisfactoria)*		

3.8.4 Diseño de caso de prueba ejecutar un EOT

Flujo central:

1- Debe existir al menos una categoría, para ello se ejecutan las funcionalidades de las HU (Ver historias de usuario de crear categoría, editar categoría, eliminar una categoría y eliminar todas las categorías).

2- Arrastrar una tarjeta del *set* de tarjeta a una categoría.

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en el escenario principal abre el ejercicio a ejecutar.	Se muestra el escenario de descripción.	*(Satisfactoria/No satisfactoria)*	
El usuario estando en el escenario ejecución ingresa las soluciones por cada participante. Selecciona la opción terminar.	Se termina el ejercicio.	*(Satisfactoria/No satisfactoria)*	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en la ventana de ejecutar ejercicio. No realiza ninguna solución y presiona el botón “terminar”.	El sistema muestra el mensaje “Debe realizar al menos una solución” y se indica el campo que debe ser llenado obligatorio.	*(Satisfactoria/No satisfactoria)*	
Evaluación de la Prueba:	*(Satisfactoria/No satisfactoria)*		

3.8.5 Diseño de caso de prueba ejecutar un EAS

Flujo central:

1- Arrastrar una etiqueta del *set* de etiquetas a una posición

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en el escenario principal abre el ejercicio a ejecutar.	Se muestra el escenario de descripción.	*(Satisfactoria/No satisfactoria)*	
El usuario estando en el escenario ejecución ingresa las soluciones por cada participante. Selecciona la opción terminar.	Se termina el ejercicio.	*(Satisfactoria/No satisfactoria)*	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en la ventana de ejecutar ejercicio. No realiza ninguna solución y presiona el botón “terminar”.	El sistema muestra el mensaje “Debe realizar al menos una solución” y se indica el campo que debe ser llenado obligatorio.	*(Satisfactoria/No satisfactoria)*	
Evaluación de la Prueba:	*(Satisfactoria/No satisfactoria)*		

3.8.6 Diseño de caso de prueba evaluar un EOT

Flujo central:

1. Se selecciona al menos un algoritmo de evaluación o todos
2. Se selecciona el ícono de evaluar
3. Se muestra la ventana con la gráfica de evaluación por cada uno de los algoritmos y la matriz de ocurrencia de las soluciones

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en el escenario evaluación decide importar soluciones seleccionando luego el ícono de evaluar.	Se muestra la ventana con la gráfica de evaluación por cada uno de los algoritmos escogidos.	*(Satisfactoria/No satisfactoria)*	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en el escenario evaluación decide importar soluciones seleccionando luego en el ícono de evaluar.	No se muestra la ventana con la gráfica de evaluación por cada uno de los algoritmos escogidos.	*(Satisfactoria/No satisfactoria)*	
El usuario selecciona un archivo no válido.	Mostrar un mensaje de error, especificándole al usuario el error que se ha encontrado.	*(Satisfactoria/No satisfactoria)*	
Evaluación de la Prueba:	*(Satisfactoria/No satisfactoria)*		

3.8.7 Diseño de caso de prueba evaluar un EAS

Flujo central:

1. Se selecciona el ícono de evaluar
2. Se muestra la ventana con la gráfica de evaluación y la tabla de ocurrencia de las soluciones.

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en el escenario evaluación decide importar soluciones dando luego clic en el ícono de evaluar.	Se muestra la gráfica y la tabla de ocurrencia de soluciones.	*(Satisfactoria/No satisfactoria)*	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
El usuario estando en el escenario evaluación decide importar soluciones dando luego clic en el ícono de evaluar.	No se muestra la gráfica y la tabla de ocurrencia de soluciones.	*(Satisfactoria/No satisfactoria)*	
El usuario selecciona un archivo no válido.	Mostrar un mensaje de error, especificándole al usuario el error que se ha encontrado.	*(Satisfactoria/No satisfactoria)*	
Evaluación de la Prueba:	*(Satisfactoria/No satisfactoria)*		

3.9 Aporte de la solución

La solución propuesta permite a los arquitectos de información agilizar la aplicación de la técnica ordenamiento de tarjetas y del método de análisis de secuencia, posibilita su uso por encima de las herramientas privativas que existen en el mundo que automatizan la técnica de ordenamiento de tarjetas e incluye el método de análisis de secuencia. El uso de la misma eleva grandemente la calidad

de los productos con la aplicación de la arquitectura de información, pues el análisis de los resultados de forma manual puede provocar errores humanos, por la complejidad de los algoritmos, que incidirían directamente en la calidad de los resultados obtenidos. La herramienta posibilita reducir estos errores al automatizar dichos algoritmos. Ayuda a la toma de decisiones de los profesionales de la arquitectura de la información en la representación de contenidos en los productos a desarrollar.

3.10 Conclusiones del capítulo.

Con la realización de este capítulo se concluye que:

- Los requisitos obtenidos en el capítulo anterior se encuentran bien descritos y satisfacen las necesidades del cliente.
- Los agrupamientos obtenidos de la aplicación k-sort fueron comparados con los agrupamientos obtenidos en la herramienta matemática R arrojando los mismos resultados.
- La definición de los patrones de diseños y de arquitectura mejoran la calidad del diseño y la implementación.
- La descripción de la arquitectura permite un mejor entendimiento del sistema a los desarrolladores.
- El diseño de casos de pruebas permitirá corregir errores que puedan existir en el sistema.

Conclusiones generales

- El estudio de disímiles herramientas que automatizan la técnica de ordenamiento de tarjetas, demuestra que ninguna de las identificadas automatiza el método de análisis de secuencia.

- Se comprobaron los agrupamientos obtenidos de la aplicación k-sort con la herramienta matemática R arrojando los mismos resultados, lo que valida la utilización de estos.

- La solución propuesta permite apoyar la toma de decisiones de los arquitectos de información a partir del análisis que se obtiene de los agrupamientos y la organización secuencial de los contenidos a representar en una solución informática desde el punto de vista del usuario.

Recomendaciones

- Integrarle a la aplicación K-sort los algoritmos de la herramienta matemática R para obtener una mayor gama de evaluaciones de los agrupamientos.

- Modelar una herramienta web que automatice el proceso de ejecución de la técnica de ordenamiento de tarjetas y el método de análisis de secuencia para ser usada por los arquitectos de información en las instituciones que cuenten con una red de computadoras disponibles.

Bibliografía

Bibliografía Referenciada

Blanco Carlos, Ruiz Francisco. 2006. Técnicas para capturar requisitos. Universidad de Cambria, 2006.

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

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

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

Optimal Workshop. 2010. Optimal Workshop. [En línea] 2010 http://www.optimalworkshop.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.

Pavón, Juan. 2008. El patrón Modelo-Vista-Controlador (MVC). Madrid: Artificial Universidad Complutense. Departamento. Ingeniería del *Software* e Inteligencia, 2008.

Peñalver, G., Meneses, A., García, S. 2010.*SXP, Metodología ágil para el desarrollo de software.* La Habana: Universidad de las Ciencias Informáticas, 2010.

Pérez-Montoro, Mario. 2010. Arquitectura de la Información en entornos web, 2010.

Pressman. 1998. [En Línea] http://www.buenastareas.com/ensayos/Estrategias-De-Pruebas-De-Software-Convencionales/1045971

Ronda León, Rodrigo. 2007. [En línea] http://www.nosolousabilidad.com/articulos/tecnicas_ai.htm

Ronda León, Rodrigo y Mesa Rábade, Yaima. 2005 [En línea] http://www.nosolousabilidad.com/articulos/analisis_secuencia.htm

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.

Bibliografía

Universidad de Salamanca. 1999. Guía de Iniciación al Lenguaje JAVA [En línea] 10, 1999. http://zarza.usal.es/~fgarcia/doc/tuto2/Index.htm.

***Vicente, Villardón**, José Luis. INTRODUCCIÓN AL ANÁLISIS DE CLÚSTER. Departamento de Estadística Universidad de Salamanca: s. n. 9.*

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.

WebSort. 2009. WebSort. [En línea] 2009. http://uxpunk.com/websort

XSort. 2011. XSort. [En línea] 2011. http://www.xsortapp.com/

Bibliografía Consultada

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

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/

Blanco Carlos, Ruiz Francisco. 2006. Técnicas para capturar requisitos. Universidad de Cambria, 2006.

Brito Acuña, Karenny. 2009. [En Línea] http://www.eumed.net/libros/2009c/584/index.htm

Celis, Ismael. 2005. [En línea] http://www.webtaller.com/maletin/articulos/el-ataque-de-los-frameworks.php

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

Espinosa Ramírez, Jiménez Morales. 2010. Implementación del módulo “Técnica de ordenamiento de tarjetas” para la plataforma de arquitectura de la información “Abad”. La Habana: Universidad de las Ciencias Informáticas, 2010.

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

Hassan, Y, Martín, F.J. 2004. Arquitectura de la Información en los entornos virtuales de aprendizaje: Aplicación de la técnica de *Card Sorting* y análisis cuantitativo de los resultados, 2004.

Bibliografía

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/

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/

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

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

Optimal Workshop. 2010. Optimal Workshop. [En línea] 2010 http://www.optimalworkshop.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.

Owen Martin, Raj, Jog. 2003. *"BPMN and Business Process Management". 2003.*

Pavón, Juan. 2008. El patrón Modelo-Vista-Controlador (MVC). Madrid: Artificial Universidad Complutense. Departamento. Ingeniería del *Software* e Inteligencia, 2008.

Peñalver, G., Meneses, A., García, S. 2010.*SXP, Metodología ágil para el desarrollo de software.* La Habana: Universidad de las Ciencias Informáticas, 2010.

Pérez-Montoro, Mario. 2010. Arquitectura de la Información en entornos web, 2010.

Pressman. 1998. [En Línea] http://www.buenastareas.com/ensayos/Estrategias-De-Pruebas-De-Software-Convencionales/1045971

Bibliografía

Prieto, Félix. 2008. Patrones de diseño. Universidad de Valladolid. Departamento de Informática, 2008.

Ronda León, Rodrigo. 2007. [En línea] http://www.nosolousabilidad.com/articulos/tecnicas_ai.htm

Ronda León, Rodrigo y Mesa Rábade, Yaima. 2005 [En línea] http://www.nosolousabilidad.com/articulos/analisis_secuencia.htm

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.

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

Universidad de Salamanca. 1999. Guía de Iniciación al Lenguaje JAVA [En línea] 10, 1999. http://zarza.usal.es/~fgarcia/doc/tuto2/Index.htm.

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

Vicente, José Luis. 2004. *INTRODUCCIÓN AL ANÁLISIS DE CLÚSTER. Departamento de Estadística Universidad de Salamanca: s.n. 9, 2004.*

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.

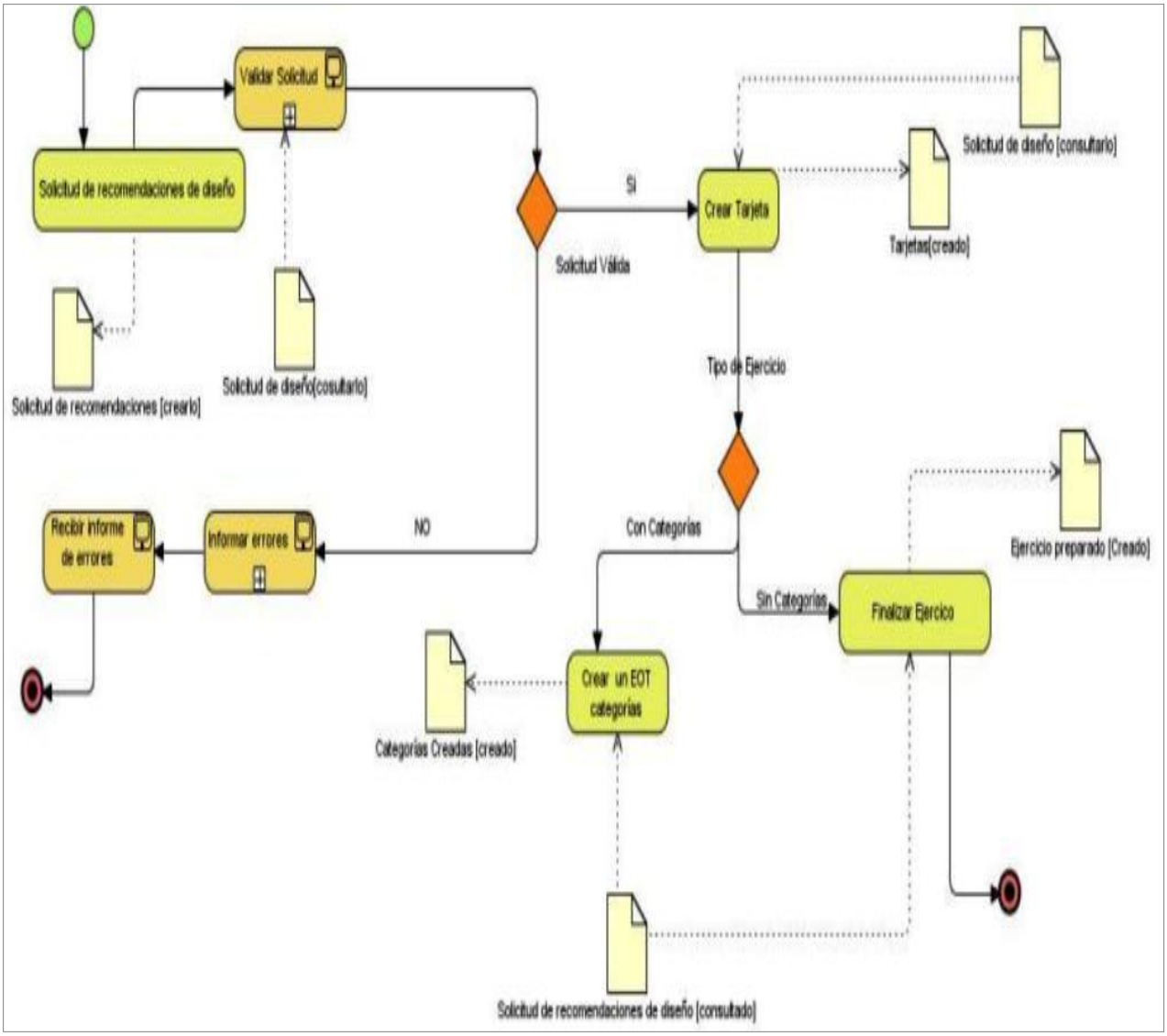
WebSort. 2009. WebSort. [En línea] 2009. http://uxpunk.com/websort

Wiegers, K. E. 2009. *Software Requirements,* Redmond, WA: Microsoft Press, 2009.

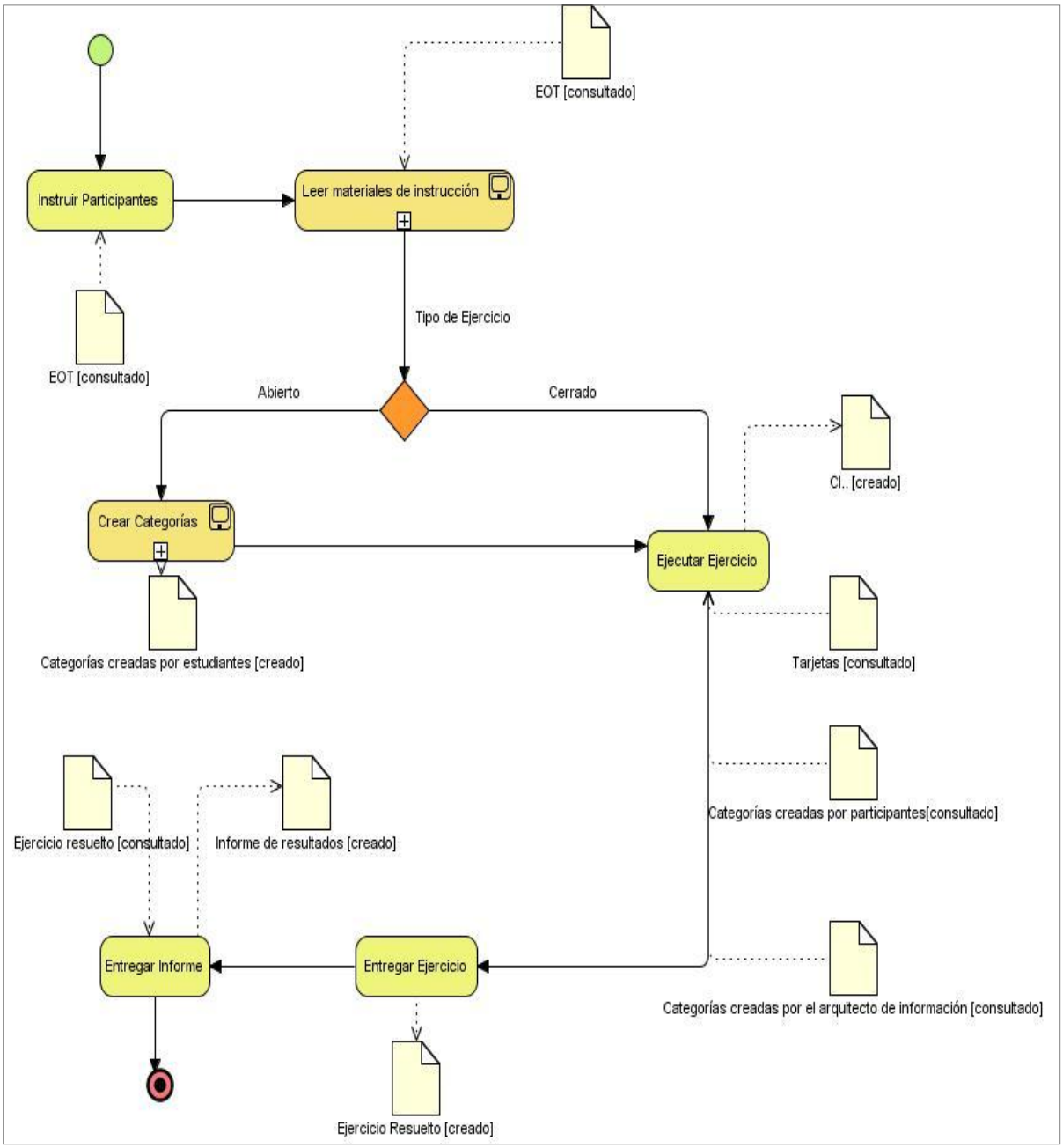
Wurman, Richard Saul. 1997. Information Architecture. Los Ángeles: Watson-Guptill Pubis, 1997.

XSort. 2011. XSort. [En línea] 2011. http://www.xsortapp.com/

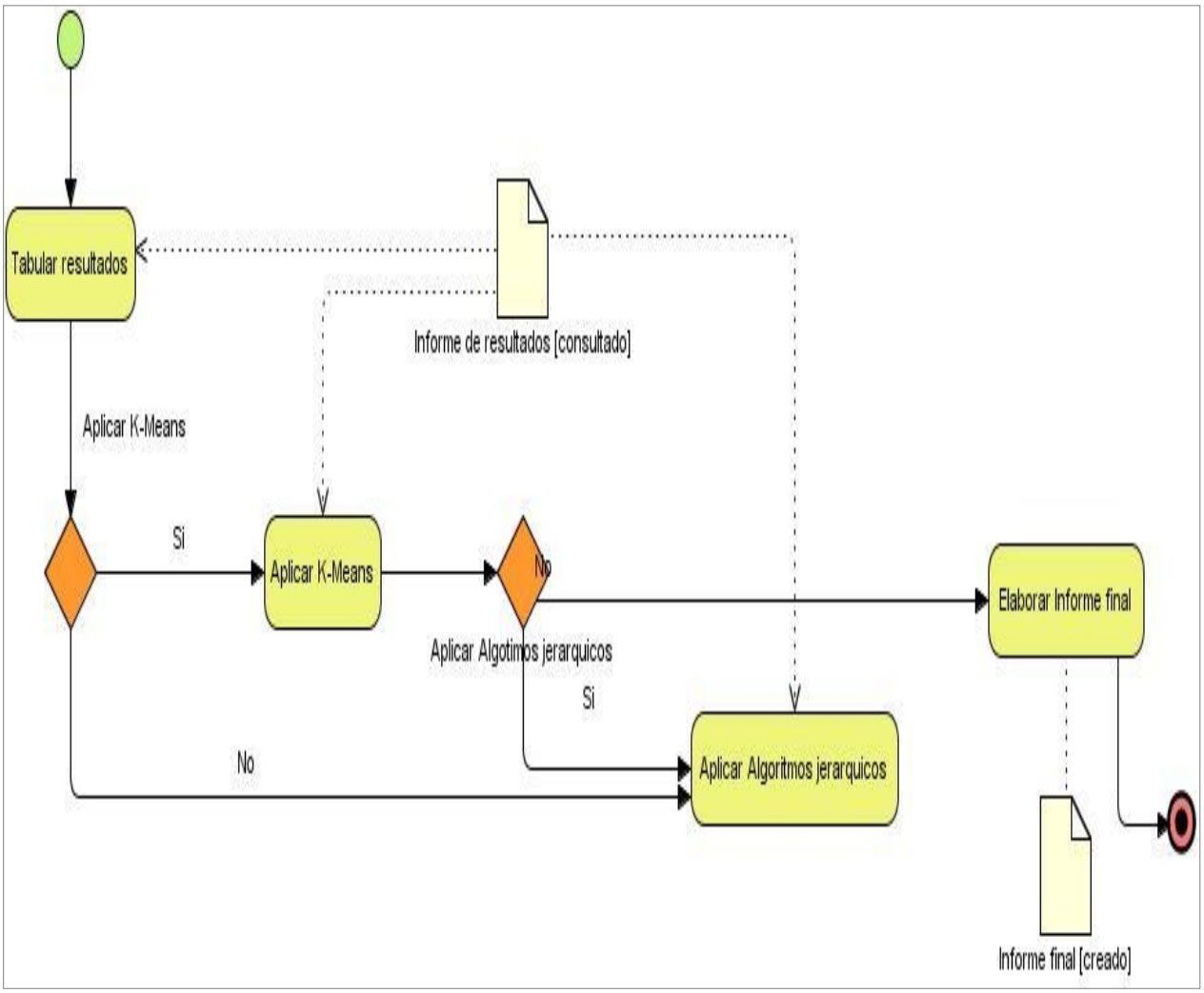
Anexo 1 Proceso de preparación de un ejercicio de ordenamiento de tarjeta



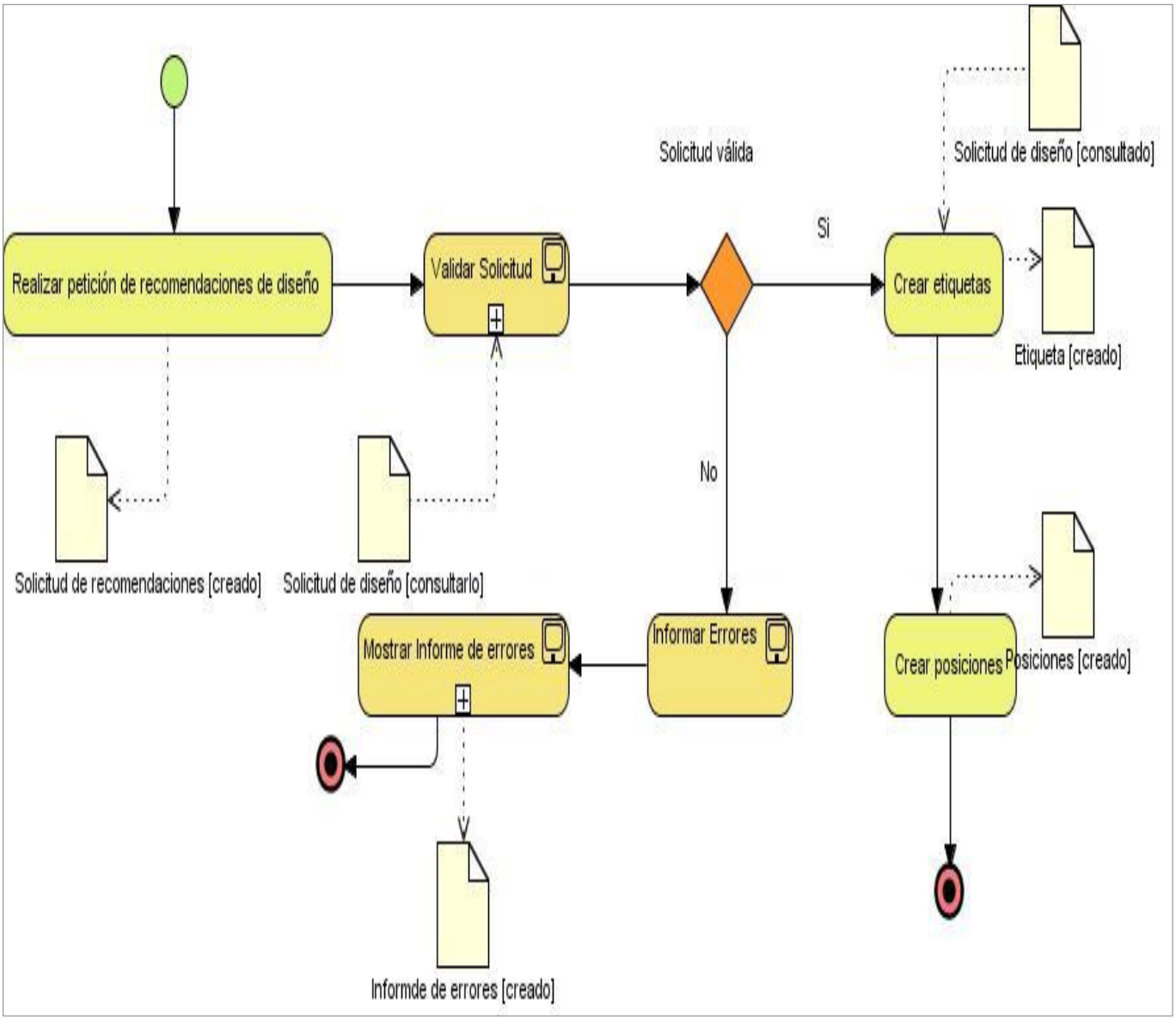
Anexo 2 Proceso de ejecución de un ejercicio de ordenamiento de tarjeta



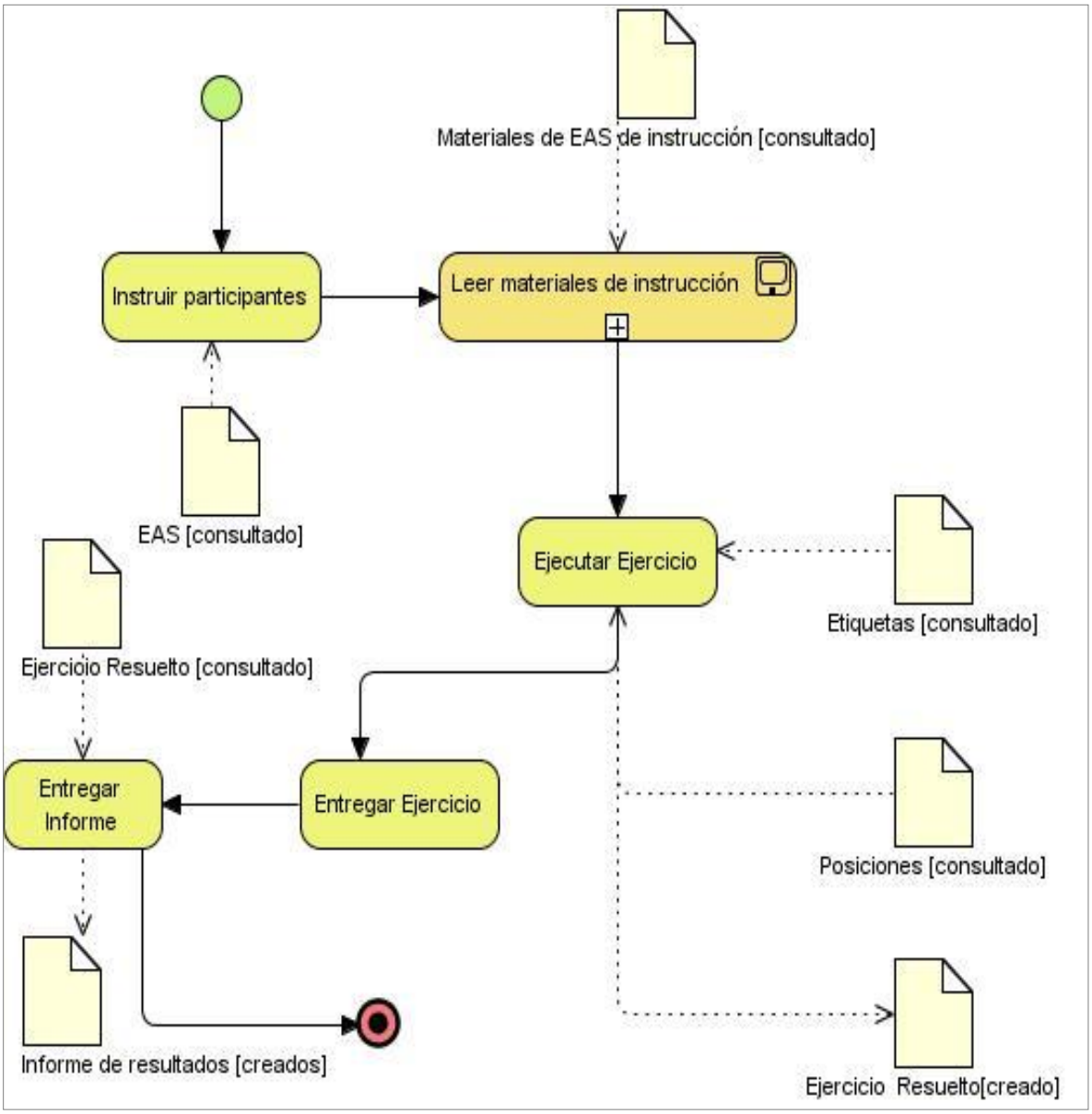
Anexo 3 Proceso de evaluación de un ejercicio de ordenamiento de tarjeta



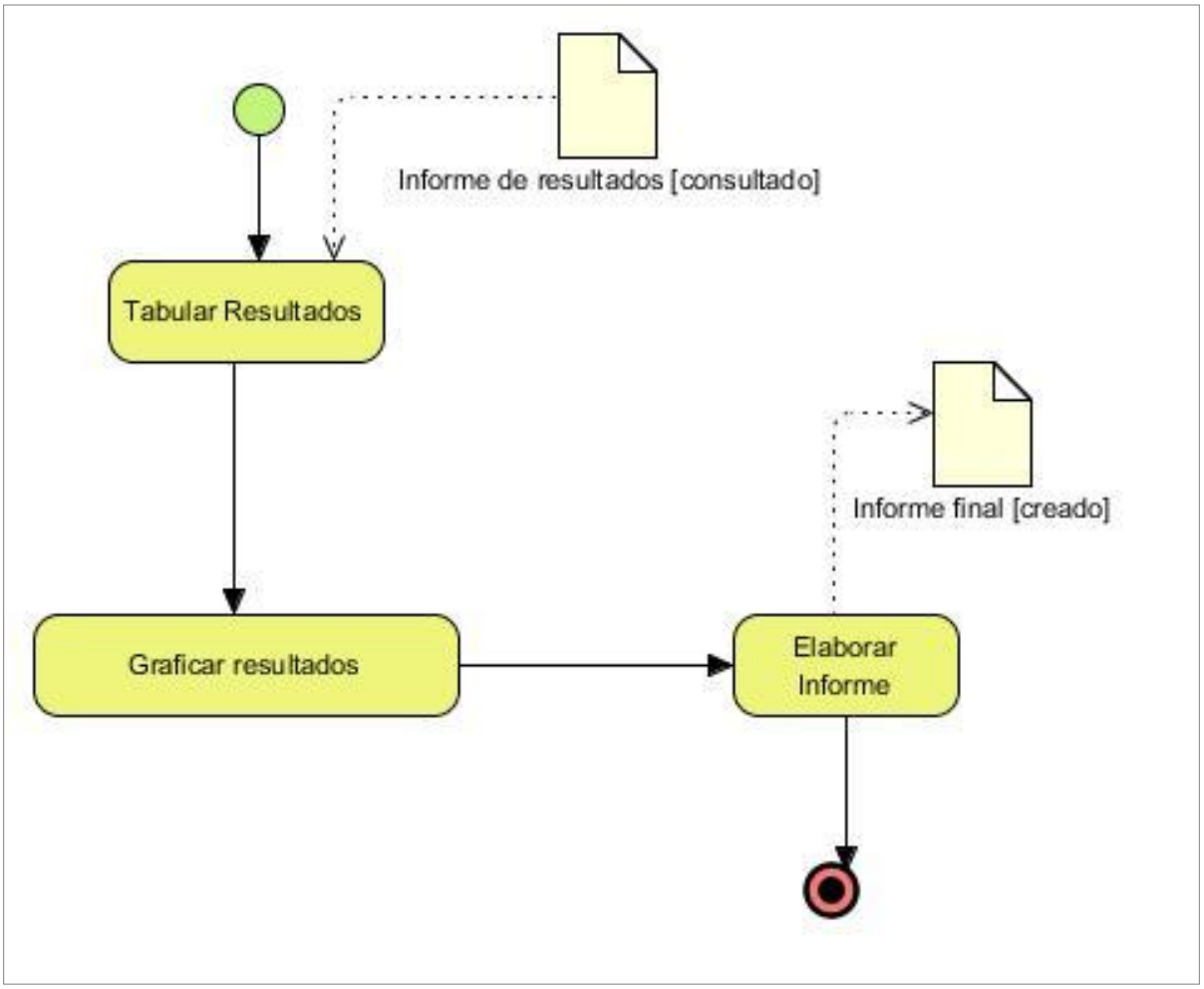
Anexo 4 Proceso de preparación de un ejercicio de análisis de secuencia



Anexo 5 Proceso de ejecución de un ejercicio de análisis de secuencia

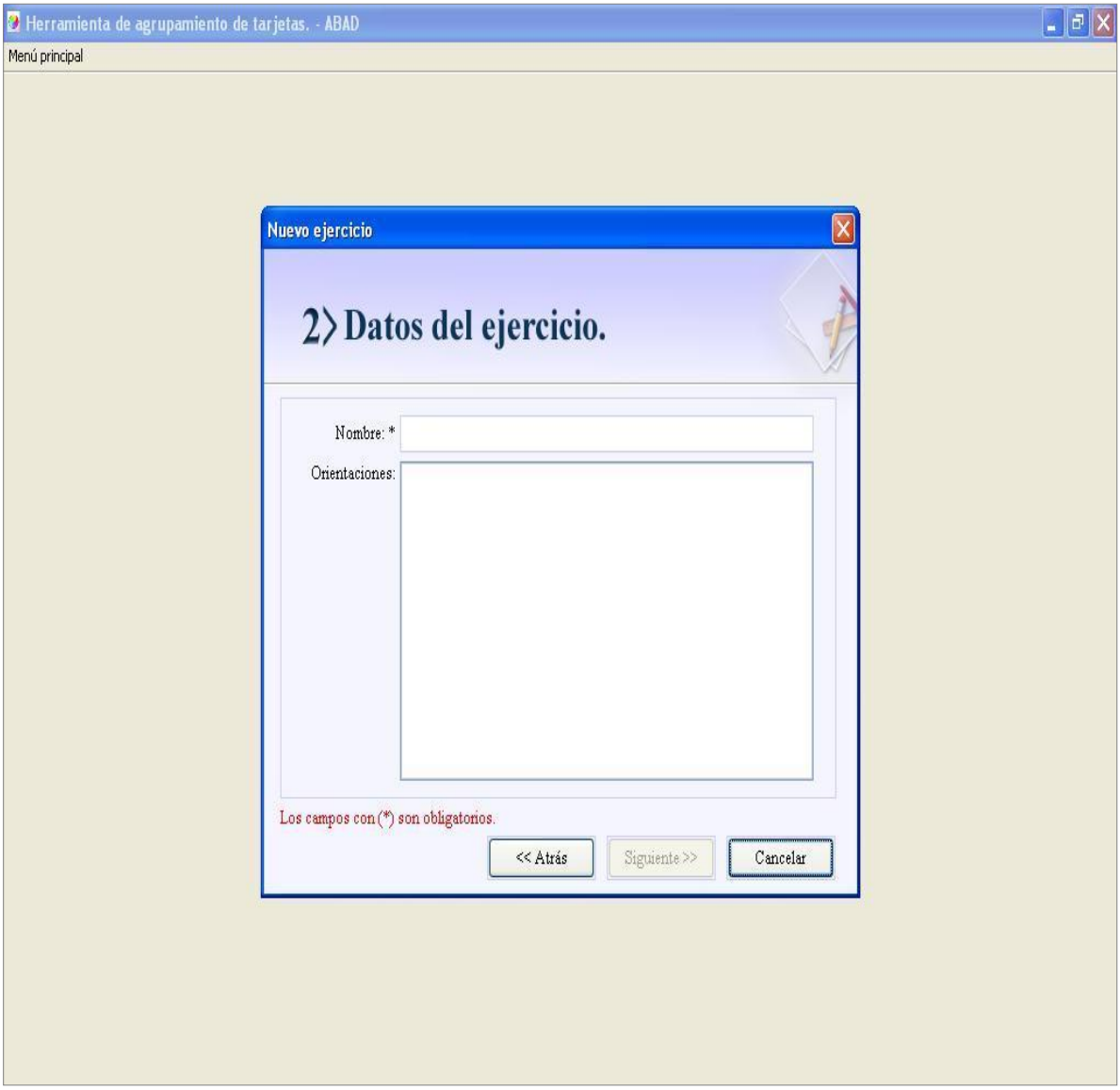


Anexo 6 Proceso de evaluación de un ejercicio de análisis de secuencia



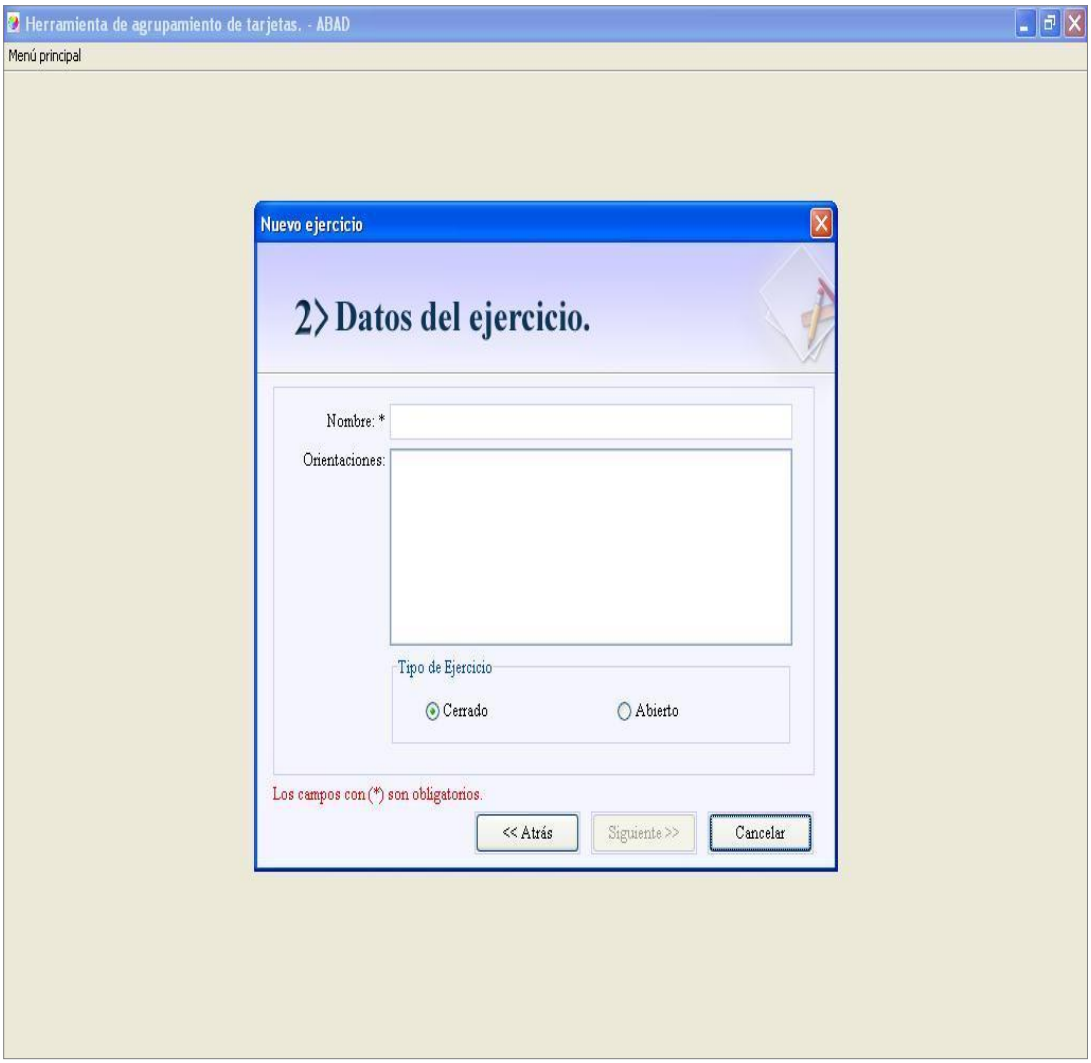
Anexo 7 Historia de usuario crear ejercicio

Pantalla: Datos del ejercicio de análisis de secuencia.



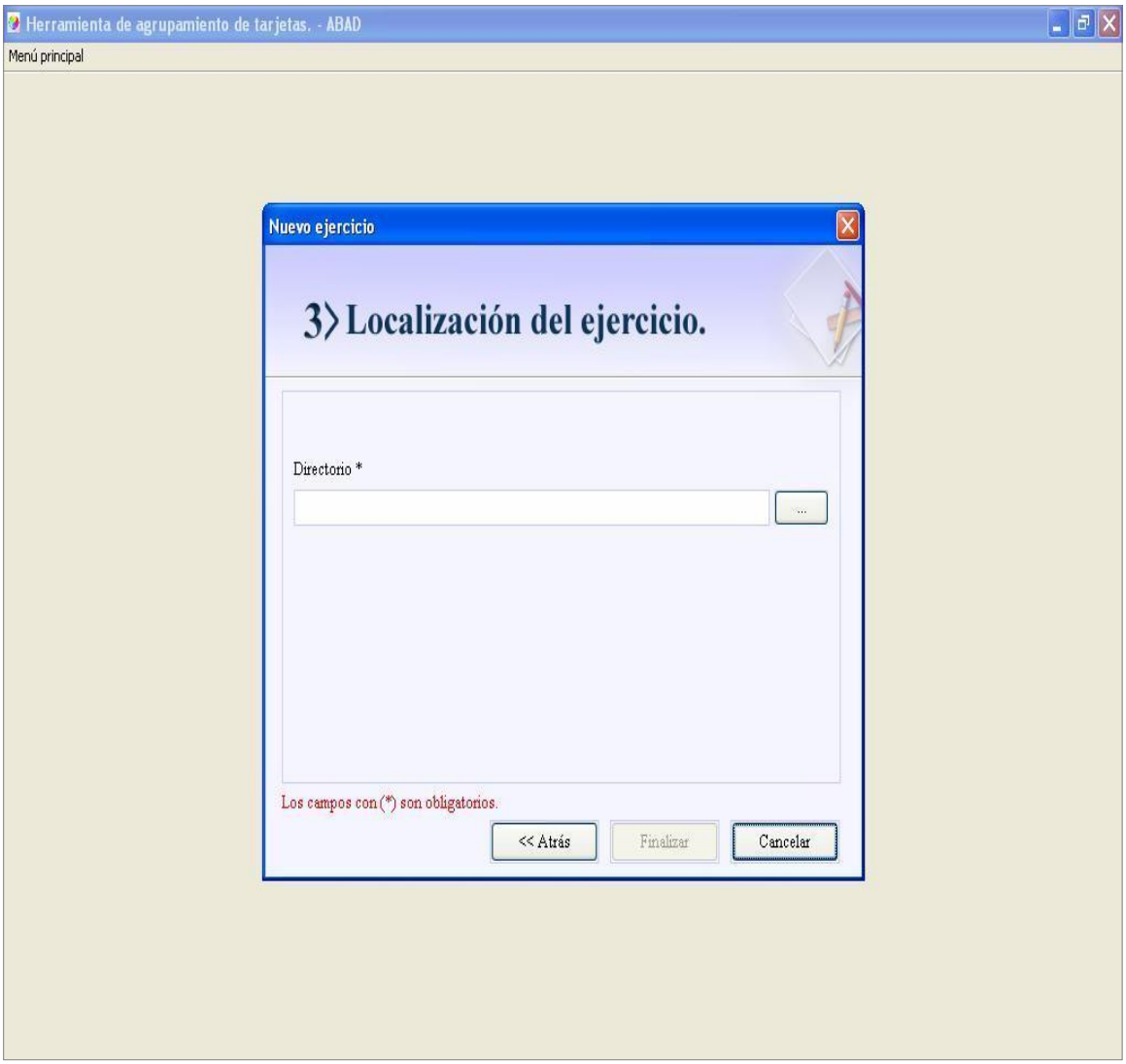
Anexo 8 Historia de usuario crear ejercicio

Pantalla: Datos del ejercicio de ordenamiento de tarjetas.



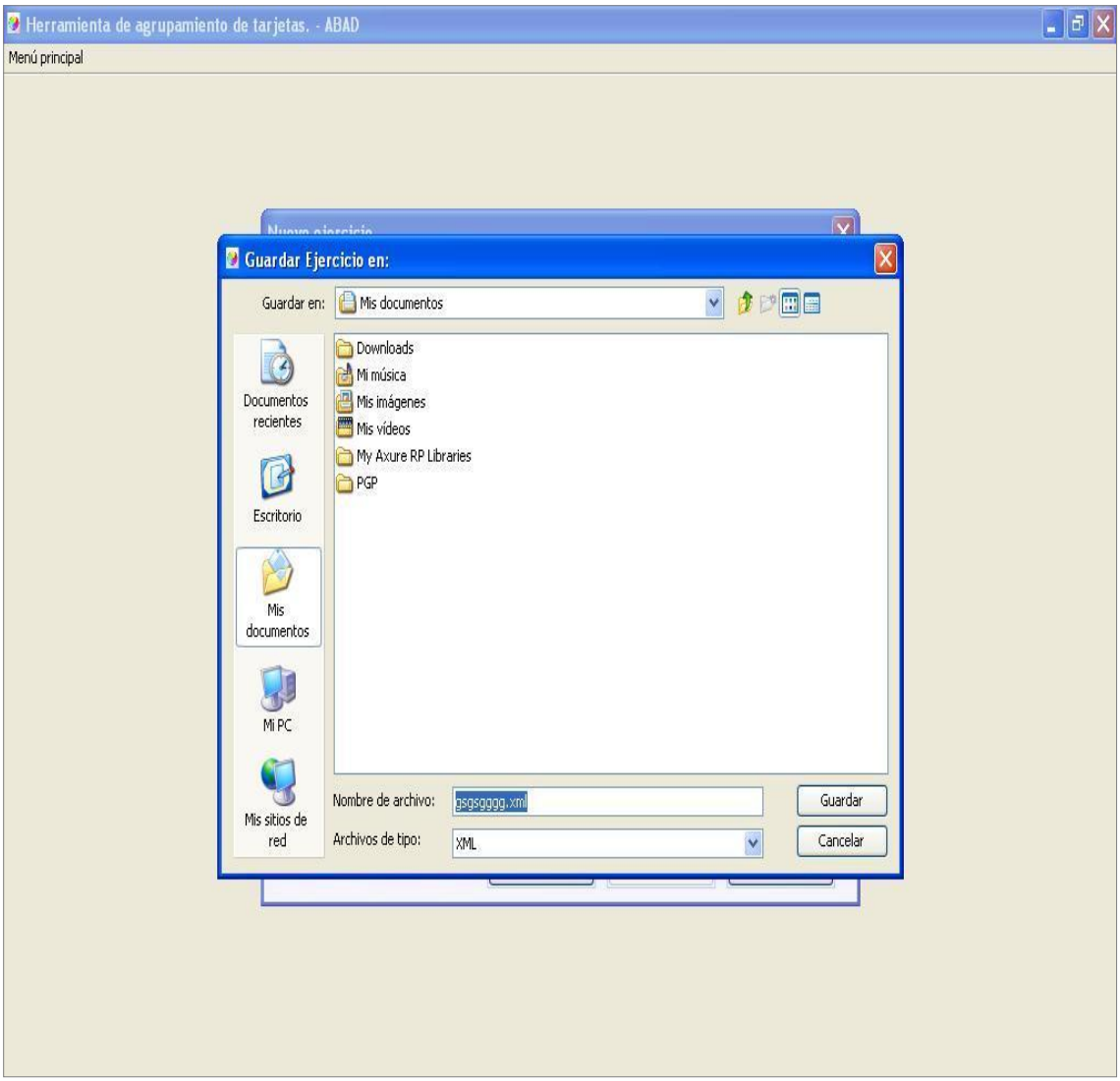
Anexo 9 Historia de usuario crear ejercicio

Pantalla: Localización del ejercicio.

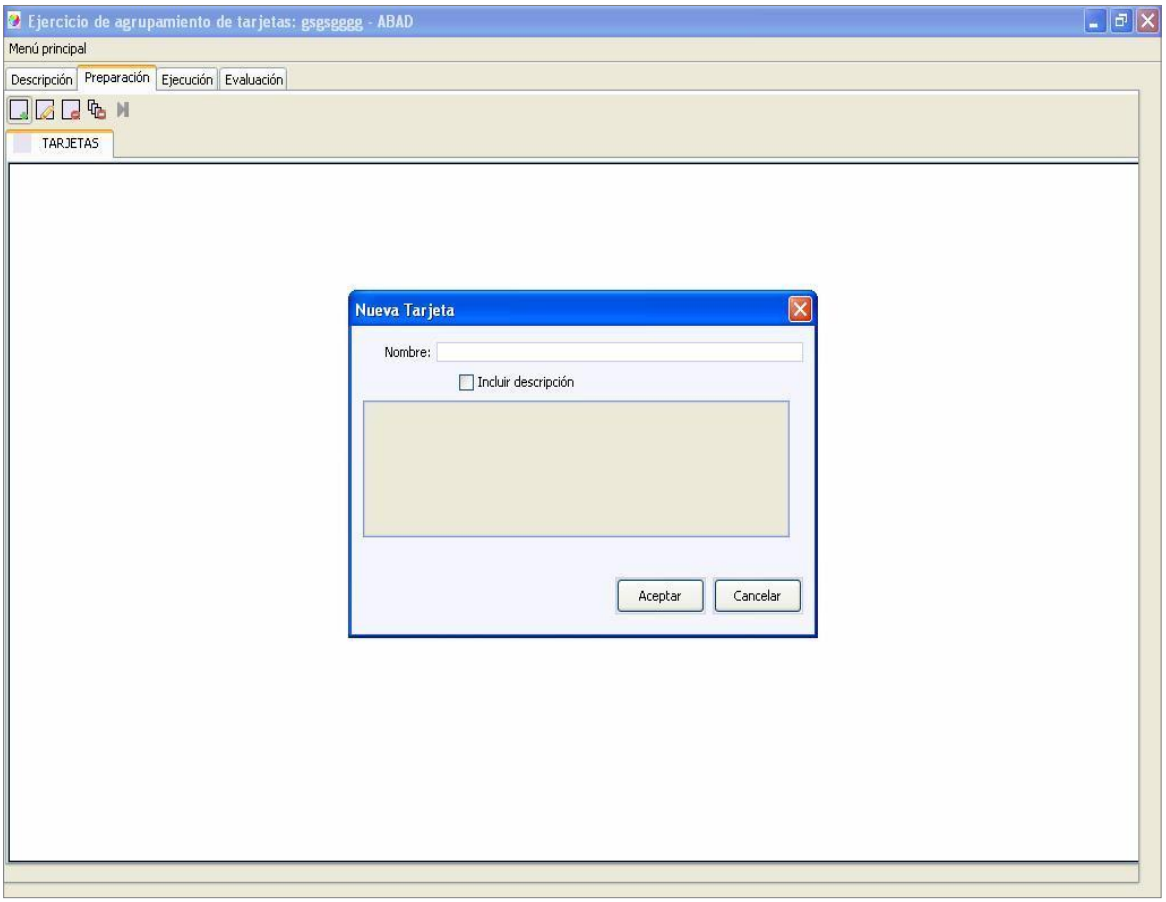


Anexo 10 Historia de usuario crear ejercicio

Pantalla: Localización del ejercicio.



Anexo 11 Historia de usuario crear una tarjeta en un ejercicio de ordenamiento de tarjeta



Anexo 12 Historia de usuario evaluar un ejercicio de ordenamiento de tarjeta