FACULTAD 1

CENTRO DE INFORMATIZACIÓN UNIVERSITARIA

HERRAMIENTA PARA LA EJECUCIÓN DE LA TÉCNICA TEST DE USUARIO EN EL ANÁLISIS DE USABILIDAD

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

Autor:

José Miguel Zaldivar Alfonseca

Tutores:

MSc. Yanicet Aveleira Rodríguez

Ing. Sergio René Vázquez Rodríguez

La Habana, Junio del 2013
"Año 55 de la Revolución"


No se vive
celebrando victorias,
sino superando
derrotas.

Agradecimientos

A mi madre que amo tanto, por todo su apoyo y comprensión ya que sin ella hoy no sería lo que soy.

A mi hermano Juan Guillermo y a mi cuñada Marbelis por estar siempre a mi lado y apoyarme cuando más los necesitaba.

A mis sobrinos Juan Alberto y Luis Enrique porque más que mi sobrino son como hermanos para mí.

A Mireya, Susana y Yaremi, por estar siempre al tanto de mí, por apoyarme y por tener un lugar importante en mi corazón.

A mis tutores Yanicet y Sergio René, por ser esenciales en la realización de este trabajo y durante mi desempeño en el proyecto.

A mis amigos Liván, Ramón, Mayelin, Luis Daniel, Yudaimy, Diana, Ramniel, Osiel, Ripoll, Juan Heriberto, Víctor y a todos aquellos que estuvieron presentes en mi vida durante estos cinco años. Gracias por permitirme conocerlos y por darme su mano cuando más difícil se hizo el camino.

Dedicatoria

Dedico este trabajo

A mi madre, mi hermano, mi cuñada y mis dos sobrinos por ser la razón de mi existir, por ser las persona que siempre están conmigo en los momentos más difícil de mi vida, los amo y quiero con todo mi corazón.

Resumen

La usabilidad es un atributo que influye en la calidad del software, sin embargo es un elemento que apenas se toma en cuenta cuando se confecciona un producto. Puede ser evaluada a través de la aplicación de diferentes métodos, entre los cuales se encuentra la técnica de test de usuario. La realización de un test de usuario de forma manual puede tornarse un poco complicada, de ahí que sea haga necesario el desarrollo de aplicaciones que automaticen este proceso. Contribuyendo no solo al desarrollo de aplicaciones enfocadas en el usuario, sino también a la toma de decisiones por parte de los profesionales para confeccionar un software usable. En muchos casos las herramientas existentes en la actualidad no realizan el análisis con el nivel de acabado necesario y en otras ocasiones trabajan bajo licencias privativas.

Por todas estas razones el objetivo de esta investigación se encuentra enfocado en el desarrollo de una herramienta de análisis de usabilidad que incorpore un conjunto de métodos de medición mediante la técnica de test usuario de forma tal que sus resultados apoyen la toma de decisiones de los profesionales en el proceso de desarrollo de interfaces de usuario. En donde se realiza un análisis de aspectos teóricos conceptuales relacionados con la gestión y evaluación de usabilidad, se incluyen además las herramientas y tecnologías presentadas para la modelación e implementación de la solución y los artefactos que sustentan la propuesta.

Palabras claves: interfaces de usuario, test de usuario, usabilidad.

Índice de contenido

Índice de figuras

Índice de Tablas

Introducción

Durante las últimas décadas el desarrollo tecnológico ha modificado sustancialmente todas las actividades de la sociedad. Por lo que los avances en la tecnología han propiciado la incorporación de nuevas herramientas y han generado a su vez nuevos escenarios de acción, lográndose así que estos recursos tecnológicos estén orientados a satisfacer los deseos de los más exigentes usuarios. Atendiendo a esto se ha incrementado el interés por perfeccionar las tecnologías de la información y las comunicaciones con una serie de buenas prácticas dentro del proceso de desarrollo de software.

En el mundo existen muchos sistemas interactivos que han sido diseñados para acomodarse a los gustos del ingeniero que lo diseñó, y no de los usuarios que le van a dar uso. La informática es una herramienta que permite al ser humano hacer cosas que no se puedan realizar de forma manual o que las automatice si este fuera el caso, abriendo así un mundo de posibilidades. Pero para que resulte ser un instrumento útil debe ponerse al servicio del usuario, adecuarse a su forma de pensar y a las capacidades y limitaciones de percepción del ser humano. Debe también diseñarse de forma que los errores más comunes que se cometen sean evitados o atenuadas sus consecuencias. Así, aplicando un enfoque de diseño centrado en el usuario, se puedan conseguir tener sistemas con una buena usabilidad y hacer la vida más fácil y satisfactoria a los millones de usuarios de sistemas informáticos.

Dada esta situación se impone desarrollar sistemas donde se contemplen las exigencias y necesidades del usuario con un buen funcionamiento de la interfaz de usuario y una buena organización de la información, para obtener una correcta comprensión de los contenidos. Permitiéndole interactuar de manera sencilla, fácil y cumpliendo sus objetivos.

En la Universidad de las Ciencias Informáticas (UCI), la usabilidad es un tema en el que cual se está trabajando ya que la mayoría de los sistemas que se desarrollan prácticamente no se les aplica ningún proceso para lograr la mejora con respecto su facilidad de uso; lo cual hace que el atributo usabilidad sea incluso desconocido para algunos de los integrantes de los proyectos productivos existentes en la UCI.

Uno de los factores que influye en una mala gestión de la usabilidad en la UCI, es que los usuarios finales son mayoritariamente excluidos del desarrollo del producto de software y en ocasiones su participación es nula. Los usuarios finales son los que determinan si un sistema es fácil de usar o no, estableciendo realmente si tiene o no calidad, puesto que la usabilidad es considerado como un atributo de la calidad del software. Debe existir una interacción usuario-grupo de desarrollo-cliente desde las etapas iniciales del proyecto para así lograr una mejor calidad en las aplicaciones producidas en la Universidad.

Para favorecer el desarrollo de interfaces teniendo en cuenta la experiencia de usuario, el Centro de Informatización Universitaria (CENIA) se ha propuesto desarrollar soluciones en ese sentido. Estas

soluciones se agrupan en “Abad”, una propuesta de paquete de herramientas para apoyar el diseño de experiencia de usuario. En este momento ya existen dos de estas soluciones: una para realizar diagramas de funcionamiento (flujo), organización y prototipos de interfaz de usuarios y la otra para aplicar las técnicas de agrupamiento de tarjetas (c*ard sorting,* en inglés) y análisis de secuencia.

Luego del análisis de toda esta situación y con el propósito de continuar contribuyendo a mejorar en alguna medida las soluciones que se desarrollan en la Universidad se hace necesario plantearse nuevas tareas que lo permitan.

Es de esta forma que aparece el siguiente **problema de la investigación:** ¿Cómo apoyar la toma de decisiones en el proceso de desarrollo de interfaces de usuario contribuyendo a una mayor usabilidad de las mismas?

Para un entendimiento global del tema, se trabaja sobre el **objeto de estudio** la disciplina de Usabilidad, delimitando como **objetivo general:** Desarrollar una herramienta de análisis de usabilidad para el paquete de herramientas de diseño de experiencia de usuario (Abad) que incorpore un conjunto de métodos de medición mediante la técnica de test[1] usuario de forma tal que sus resultados apoyen la toma de decisiones de los profesionales en el proceso de desarrollo de interfaces de usuario.

De ahí que el **campo de acción** quede enmarcado en: la técnica test de usuario para medir la usabilidad. Para lograr el cumplimiento del objetivo propuesto se hace necesaria la aplicación de diferentes **métodos científicos**, los cuales son:

Métodos teóricos:

Método analítico- sintético: se utiliza para el análisis y estudio de la información recopilada, separando los elementos más relevantes para facilitar su entendimiento. Permite caracterizar, definir conceptos, términos con respecto a la integración de sistemas, también posibilita enfocar la investigación en las diferentes arquitecturas existentes para lograr el objetivo propuesto y contribuye a la selección de los diferentes lenguajes y herramientas para el desarrollo de dicha solución.

Histórico-lógico: se utiliza para realizar el estudio histórico del fenómeno, la lógica interna de su desarrollo y su teoría. Lo cual permitirá expresar en forma teórica la esencia de la integración entre sistemas, explicar la historia de su desarrollo y unir el estudio de su estructura con su concepción histórica.

[1] El Diccionario de la Real Academia Española (DRAE) desde la edición de 1992, aceptó el anglicismo test por su uso generalizado, lo cual significa que es un préstamo lingüístico del idioma inglés hacia otro idioma, en este caso el español. Según el DRAE significa “prueba destinada a evaluar conocimientos o aptitudes, en la cual hay que elegir la respuesta correcta entre varias opciones previamente fijadas”. Además este término es muy utilizado en psicología con el sentido de “prueba de carácter psicológico o psicotécnico para estudiar o evaluar una función”. Fuera de estos dos sentidos específicos, no debe emplearse este anglicismo por existir las palabras en español que lo sustituyan. Debido a que varios autores especialistas en usabilidad como Yusel Hassan, Francisco J Martín, Jakob Nielsen, Thomas G. O´Brien, entre otros, utilizan este término para nombrar la técnica desarrollar en la presente investigación, se decide la utilización de dicho anglicismo.

En fin, el uso de estos métodos posibilita extraer y diferenciar técnicas, metodologías y conceptos básicos sobre la usabilidad en los productos del software. Facilitan identificar características, según el grado de aplicación y tendencias en el mundo. Además, permiten analizar y sintetizar el objeto de estudio pues la síntesis se realiza sobre la base de los resultados previos del análisis.

Estructura Capitular

La presente investigación está compuesta por una introducción, un resumen, un índice general, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía, glosario de términos y anexos. Los 3 capítulos están desglosados de la siguiente forma:

Capítulo 1. Fundamentación teórica

Se realiza la elaboración del marco teórico de la investigación a partir de la definición y descripción de lo que es usabilidad, mediante las definiciones de autores reconocidos y organizaciones internacionales. Se describen los elementos de usabilidad como atributo de calidad del software y el papel de la misma en el modelo CMMI. Se realiza un estudio de las normativas UCI en cuanto al tema de análisis de usabilidad, además de cómo se gestiona la usabilidad atendiendo los criterios del diseño centrado en el usuario y la ingeniería de usabilidad. Se trata todo lo referente al test de usuario para el análisis de usabilidad y se describen los tipos de test de usuario. Se realiza un estudio del estado del arte sobre soluciones la existencia de herramientas para realizar test de usuario en la UCI, Cuba y el mundo. Además, se define lenguaje de programación, marco de trabajo, metodología de desarrollo, ambiente de desarrollo, patrones a utilizar.

Capítulo 2. Implementación de la solución propuesta

Se realiza una valoración crítica del análisis y se caracteriza el sistema propuesto por la analista. Se describe el diseño de la propuesta solución. Se realiza la selección de los patrones arquitectónicos y de diseño a utilizar. Se define el estándar de codificación a utilizar. Se realiza la modelación del sistema, a través del diagrama de paquetes, de implementación y despliegue.

Capítulo 3. Validación de la solución propuesta

Se aborda el tema referente a la realización de las pruebas; se realiza un análisis del tipo de prueba a utilizar para validar el software y el diseño de casos de prueba. En este capítulo también se crean los casos de prueba para cada funcionalidad que se han implementado donde se detallan las mismas con sus descripciones generales, condiciones de ejecución, y secciones a probar. Se realiza además el análisis de los resultados obtenidos al finalizar las iteraciones de pruebas.

Capítulo 1: Fundamentación teórica

Introducción

En el presente capítulo se realiza la elaboración del marco teórico de la investigación a partir de la definición y descripción de lo que es usabilidad, mediante las definiciones de autores reconocidos y organizaciones internacionales. Se describen los elementos de usabilidad como atributo de calidad del software y el papel de la misma en el modelo CMMI. Se realiza un estudio de las normativas UCI en cuanto al tema de análisis de usabilidad, además de cómo se gestiona la usabilidad atendiendo los criterios del Diseño Centrado en el Usuario y la Ingeniería de Usabilidad. Se trata todo lo referente al test de usuario para el análisis de usabilidad y se describen los tipos de test de usuario. Se realiza un estudio del estado del arte sobre soluciones la existencia de herramientas para realizar test de usuario en la UCI, Cuba y el mundo. Además, se define lenguaje de programación, marco de trabajo, metodología de desarrollo, ambiente de desarrollo.

1.1 Principales conceptos de la investigación.

La usabilidad es un anglicismo que apareció hace algunos años, que significa “facilidad de uso”, su origen se remonta a los años 80 para sustituir el término “amigable para el usuario” (*userfriendly* traducido al inglés), cuya connotación en ese entonces había adquirido un carácter subjetivo (BEVAN, 1991).

Existen diversas definiciones de usabilidad propuestas por algunos autores, que lo han hecho desde los diversos atributos a partir de los cuales la usabilidad puede ser evaluada, alguno de estas definiciones son:

Según ISO 9241-11 es, “La medida en la que un producto se puede usar por determinados usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso específico”(STANDARDIZATION., 1998).

Según J. Nielsen[2], “La usabilidad es un atributo relacionado con la facilidad de uso. Más específicamente, se refiere a la rapidez con que se puede aprender a utilizar algo, la eficiencia al utilizarlo, cuán memorable es, cuál es su grado de propensión al error y cuánto les gusta a los usuarios. Si una característica no se puede utilizar o no se utiliza es como si no existiera” (NIELSEN 2002).

[2] Es uno de los autores más reconocidos en el ámbito de la usabilidad, es Doctor en Ciencias de la Computación e Interfaces de Usuario. Es autor de varios libros, entre ellos: *Eyetracking Web Usability* (2009), *Prioritizing Web Usability* (2006), *Designing Web Usability* (1999), *Homepage Usability: 50 Websites Deconstructed* (2001), *CoordinatingUser Interfaces forConsistency* (2001). Sus libros son traducidos a varios idiomas y las ventas lo han convertido en un autor de *best-sellers* suficientemente reconocido.

Según J. Preece[3], “La usabilidad es el desarrollo de productos interactivos fáciles de aprender, sencillos de usar y agradables desde la perspectiva del usuario. En concreto, la usabilidad se desglosa en los siguientes objetivos: efectividad, eficiencia, seguridad, utilidad, capacidad de aprendizaje y memorabilidad” (PREECE, 2007).

Nigel Bevan[4], “La usabilidad es la facilidad de uso y la aceptabilidad de un sistema o producto para una clase particular de usuarios que llevan a cabo tareas específicas en un entorno específico" (BEVAN, 1991).

Luego del análisis de estos conceptos se puede concluir que la disciplina de la Usabilidad estudia la manera de diseñar productos para que los usuarios puedan interactuar con ellos de la forma más fácil, cómoda e intuitiva, de manera que puedan conseguir sus objetivos de una forma rápida y sencilla.

1.2 Usabilidad como atributo de calidad del software

En la actualidad, mundialmente la industria del software se ha convertido en un enorme gigante, el cual crece y se desarrolla a ritmo agitado. Todas las empresas quieren producir aplicaciones informáticas con alta calidad, en el menor tiempo posible y a costos mínimos. Aumentando así la competitividad entre ellas debido a que los clientes son cada vez más exigentes. La calidad del software desempeña un papel importante dentro del desarrollo de aplicaciones informáticas influyendo positivamente en la decisión de un cliente a la hora de escoger el producto que necesita.

Según Roger S. Pressman, la calidad del software es la: “concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente”. (PRESSMAN, 2002).

El Instituto de Ingenieros Eléctricos y Electrónicos(IEEE.) expresa que la calidad de software es: “El grado con que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario” (IEEE., 1994).

“La calidad del software es el conjunto de cualidades que lo caracterizan y que determinan su utilidad y existencia. La calidad es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad” (VEGA LEBRÚN, 2000).

---

[3] Es Decana de la Facultad de Estudios de la Información en la Universidad de Maryland. Es conocida por su trabajo sobre cómo los factores de usabilidad generan éxito en comunidades en línea. Es autora de importantes libros que han aportado en el campo de la usabilidad, entre ellos: *InteractionDesign: Beyond Human-Computer Interaction* (2007), *A Guide to Usability: Human Factors in Computing* (1993), *User Interfaces in the Real World* (2007).

[4] Es un experimentado consultor en usabilidad, Director de Desarrollo Profesional de la Asociación de Profesionales de Usabilidad y lidera la iniciativa *UPA Usability Body of Know ledge*. Participa en grupos de varias normas internacionales y ha contribuido al desarrollo de la norma ISO 13407. Fue el responsable de desarrollar el nuevo estándar de la industria para los requisitos de usabilidad. Tiene un sitio web con una buena cantidad de documentos de investigación, ponencias, artículos sobre el tema de la usabilidad.

Los atributos de calidad son características que sirven para medir el sistema. De acuerdo con el estándar ISO/IEC 9126, la usabilidad es un atributo de la calidad del software. El término es utilizado para referirse a la capacidad de un producto para ser usado fácilmente. Esto corresponde a la definición de usabilidad como parte de la calidad del software. La usabilidad tiene que ver con crear sistemas fáciles de usar. Por lo que un sistema usable es: fácil de aprender, fácil de recordar, efectivo, eficiente, y seguro de usar (Ver Figura 1).

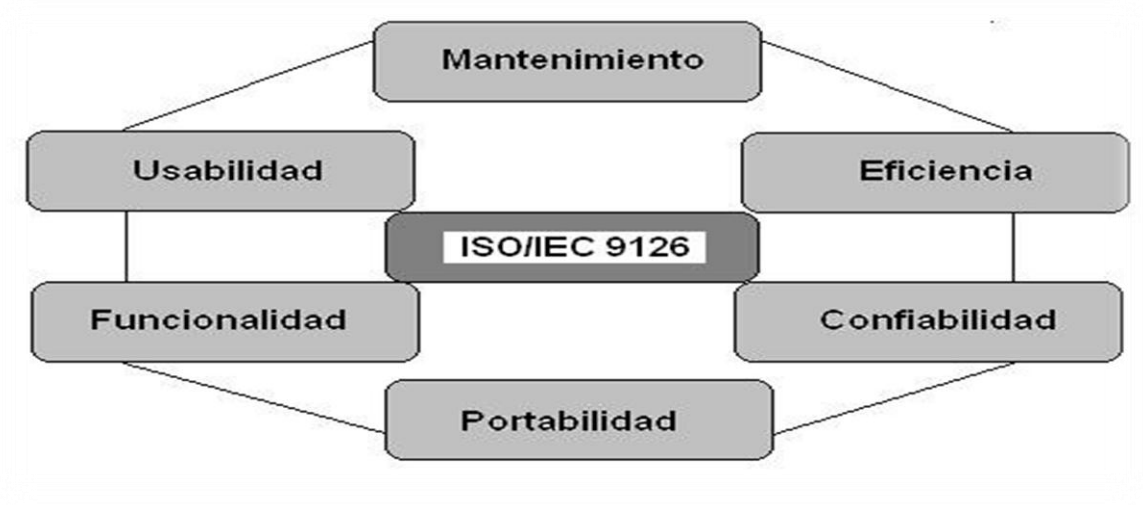


Figura 1. Atributos de la calidad según la ISO 9126.

Sobre la base de las consideraciones anteriores, se puede concluir que la calidad del software garantiza que el producto cumpla con los requisitos y especificaciones de los usuarios finales. Por lo que tal como se ha visto la usabilidad como uno de los atributos de la calidad del software, ayuda a medir la facilidad de uso que tiene cualquier producto.

1.3 La usabilidad en el modelo CMMI

Es bien conocido que en el mundo tan cambiante en el que se vive hoy en día, se necesita ser competentes. Para poder decir que nuestra empresa es competente, se tienen que utilizar a terceras personas que certifiquen qué tan bien se realiza el trabajo, y cómo quedan los productos. Es de allí entonces que nacen normas para la certificación de la calidad, y de allí es donde nace CMMI, que es una especie de manual, de qué hay que hacer para producir productos con calidad.

Representa un camino de mejoramiento, y permite determinar la madurez, y evaluar las capacidades de las organizaciones que desarrollan sistemas informáticos. Es una colección estructurada de elementos, que describe características de procesos que han demostrado, por experiencia, ser exitosos. Es recomendado para organizaciones que quieren incrementar la capacidad de su proceso de desarrollo y desarrollar productos con calidad.

El modelo de **CMMI** plantea 5 niveles de madurez. Cada nivel es un escalón bien definido de mejora de proceso y estabiliza una parte importante de los procesos organizacionales. Las empresas o instituciones que se encuentran en el **Nivel 1** de madurez según CMMI, no disponen de procesos y controles definidos. No trabajan con procedimientos que están normalizados. La planificación y el control, no están establecidos. Las técnicas y/o herramientas que se emplean para el desarrollo del sistema carecen de una integración y únicamente son empleadas en algunas fases del ciclo de vida del producto. La principal característica de las empresas que se encuentran en este nivel es que no hay un control de la gestión de proyectos de software efectivo (SHRUM, 2009).

Las empresas o instituciones que se encuentran en el **Nivel 2** de madurez según CMMI, tienen métodos estandarizados facilitando procesos repetibles. Aplican un control básico de la gestión de proyectos, gestión de calidad y gestión de la configuración. Su necesidad fundamental es establecer una administración efectiva del proyecto de software. Los procesos de administración de proyectos están definidos e implementados. Las políticas organizacionales guían los proyectos y las prácticas exitosas usadas en proyectos previos, pueden ser repetidas (SHRUM, 2009).

Uno de los aspectos que se miden en CMMI para lograr pasar al **Nivel 2** de madurez, es el aseguramiento de la calidad de productos y procesos. El aseguramiento de la calidad se enfoca en identificar y evaluar los defectos que puedan afectar al sistema. Si los errores se pueden identificar de forma temprana en el proceso de desarrollo, las características del diseño de software se pueden especificar de modo que eliminarán o controlarán los peligros potenciales, ahorrando así esfuerzos, tiempo y recursos. El aseguramiento de la calidad del software no es una tarea que se realiza en una fase particular sino durante todo el ciclo de vida del producto. Por cuanto asegurando una buena usabilidad, se garantiza la calidad del software requerida por los clientes.

1.4 Iniciativas UCI en la usabilidad

En la Universidad, algunos proyectos productivos realizan la gestión de la usabilidad a través de listas de chequeo, la cual consta de una serie de preguntas que son respondidas por el especialista de calidad en el centro Calisoft (es una organización enfocada a contribuir al desarrollo de aplicaciones informáticas en la industria cubana). Estas reciben un peso, que define si el indicador a evaluar es crítico o no; una evaluación, para dicho indicador, que toma un valor de 1 en caso de mal (cuando la respuesta al indicador sea “No”) y 0 en caso que elemento revisado no presente errores (cuando la respuesta al indicador sea “Sí”). Además de la cantidad de elementos afectados, este aspecto especifica la cantidad de errores encontrados sobre el mismo indicador.

Las listas de chequeo miden aspectos que son de interés para evaluarla usabilidad, por ejemplo que exista una buena visibilidad del sistema, un lenguaje común entre el sistema y el usuario, una libertad y control por parte del usuario de navegar en el sistema, una consistencia y estándares definidos,

prevención de errores. En ellas se analiza además que el sistema sea fácil de entender y memorizar, flexible y eficiente a la hora de usar, estético y con un diseño sin excesos, que permita al usuario reconocer, diagnosticar, recuperarse de errores que puedan surgir en el sistema.

Resulta oportuno destacar que aunque estas listas de chequeo evalúan en una medida la usabilidad que tiene un determinado sistema, todo este proceso es realizado por especialistas de calidad, nunca por usuarios reales. Esto trae por consiguiente que aunque existan aspectos en común de interés para usuarios y especialistas, pueden surgir otros aspectos importantes para tener una buena usabilidad por parte del cliente.

1.5 Gestión de la usabilidad

La gestión de la usabilidad se ha ido expandiendo, dado que la usabilidad es la cualidad de los productos que se pretende obtener mediante el Diseño Centrado en el Usuario (DCU); cuyo objetivo principal es obtener productos más usables. Igualmente se considera la ingeniería de usabilidad**,** la cual tiene el mismo propósito y es equivalente en la práctica al DCU. A continuación se detallarán en qué consisten cada uno de estos conceptos y su importancia para la gestión de la usabilidad.

1.5.1 Diseño Centrado en el Usuario

El Diseño Centrado en el Usuario se caracteriza por asumir que todo el proceso de diseño y desarrollo de un sistema debe estar conducido por el usuario, sus necesidades, características y objetivos. Centrar el diseño en los usuarios implica involucrar desde el comienzo a los usuarios en el proceso de desarrollo del producto; conocer cómo son, qué necesitan, para qué usan el sistema; testear el software con los propios usuarios; investigar cómo reaccionan ante el diseño, cómo es su experiencia de uso; e innovar siempre con el objetivo claro de mejorar la experiencia del usuario (HASSAN MONTERO 2004).

Muy simplificadamente el Diseño Centrado en el Usuario es como la práctica de diseñar productos de forma que sus usuarios puedan servirse de ellos con un mínimo de estrés y un máximo de eficiencia (WOODSON, 1981).

En resumen, podría definirse que el Diseño Centrado en el Usuario es una filosofía de diseño que tiene por objetivo la creación de productos que resuelvan necesidades concretas de sus usuarios finales, consiguiendo la mayor satisfacción y mejor experiencia de uso posible con el mínimo esfuerzo de su parte.

DCU es un proceso de diseño de interfaces de usuario en el cual se le da mucha atención a las necesidades, deseos y limitaciones de los usuarios del producto diseñado en cada estado del proceso de diseño. Se caracteriza por ser un proceso de múltiples etapas para resolver problemas que no solo requieren que los diseñadores analicen y prevean cómo los usuarios van a usar el sistema, sino también que prueben y validen sus suposiciones con respecto al comportamiento y experiencia del usuario en pruebas reales de uso por parte de este. Dichas pruebas son necesarias ya que normalmente es muy

difícil para los diseñadores de un producto imaginar la reacción que tendrán los usuarios ante el sistema diseñado y cuál será el nivel de aprendizaje de forma individual (PROJECT, 2003).La principal diferencia con otras filosofías de diseño de productos es que el DCU trata de optimizar el producto con respecto a cómo los usuarios pueden, quieren y necesitan usar el producto, en vez de forzarlos a cambiar su comportamiento normal para adaptarse al producto. Es decir, el DCU busca adaptar el producto al usuario y no al revés.

La ISO 9241-210 describe **seis principios claves** que caracterizan un DCU:

- El diseño está basado en una comprensión explícita de usuarios, tareas y entornos.
- Los usuarios están involucrados durante el diseño y el desarrollo.
- El diseño está dirigido y refinado por evaluaciones centradas en usuarios.
- El proceso es iterativo.
- El diseño está dirigido a toda la experiencia del usuario.
- El equipo de diseño incluye habilidades y perspectivas multidisciplinares (STANDARDIZATION., 2010).

El DCU ayuda a los diseñadores de software a cumplir los objetivos de un producto desarrollado para sus usuarios. Los requerimientos de los usuarios son considerados desde el principio del proceso e incluidos en todo el ciclo de desarrollo del producto. Estos requerimientos son refinados a través de varios métodos: estudios etnográficos, investigación del contexto, pruebas con prototipos, pruebas de usabilidad y otros métodos.

El DCU de sistemas interactivos puede regirse por muchos y muy diversos principios entre los que se encuentran (PROJECT, 2003):

- Diseño para los usuarios y sus tareas.
- Consistencia.
- Diálogo simple y natural.
- Reducción del esfuerzo mental del usuario.
- Proporcionar realimentación adecuada.
- Proporcionar mecanismos de navegación adecuados.
- Dejar que el usuario dirija la navegación.
- Presentar información clara.
- El sistema debe ser amigable.

- Reducir el número de errores.

De los anteriores planteamientos se deduce que el DCU está dirigido por el usuario, está enfocado a la solución que satisfaga las necesidades de este. Además, está enfocado apoyar que se cumplan con los atributos de usabilidad: efectividad, eficiencia y satisfacción. Busca una calidad de uso, de ahí que las soluciones se implementan a partir de validaciones de usuarios. De manera que el DCU persigue alcanzar en los productos la usabilidad. Por tanto, la usabilidad es la etapa más importante dentro de la forma de trabajar del DCU.

1.5.2 Ingeniería de Usabilidad

Según Nielsen, la Ingeniaría de Usabilidad da pautas para obtener productos con un alto grado de usabilidad, mediante la integración de distinto métodos del Diseño Centrado en el Usuario en los procesos del ciclo de vida del desarrollo del software de una forma estructurada y sistemática (NIELSEN, 1993). Asimismo Rosson y Carroll plantean que consiste en un conjunto de conceptos y técnicas para planificar, realizar y verificar los objetivos de usabilidad de un sistema (CARROLL., 2002).

La aplicación de Ingeniería de Usabilidad ayuda a que el software tenga una mayor calidad, que tenga mejor aceptación por los usuarios y que el producto final tenga los mejores resultados posibles. De esta forma, podría decirse que la Ingeniería de usabilidad es una metodología que proporciona la manera de proceder organizadamente para incluir la usabilidad en el desarrollo de aplicaciones interactivas.

La Ingeniería de Usabilidad tiene diferentes técnicas, las cuales buscan alcanzar un mejor nivel de usabilidad en el producto desarrollado. Para ello, antes de comenzar con el proyecto, se confecciona una lista de especificaciones de usabilidad pretendiendo plasmar los niveles de usabilidad que interesen alcanzar. Estas orientarán el proceso de desarrollo continuo, pero para fijarlas resulta necesario reconocer previamente a los usuarios y las tareas que van a realizar con el sistema. Luego de realizar el análisis de tareas, y sabiendo cuáles van a ser las soportadas por el sistema, se comienza con el diseño de la interacción del sistema, como una primera aproximación que será evaluada y, posteriormente, mejorada con iteraciones de dicho proceso. Esta evaluación permite determinar el nivel de usabilidad que el prototipo actual del sistema alcanza, y así poder identificar los defectos de usabilidad que este presenta. Existen diferentes maneras de realizar la evaluación, una de ellas es mediante el test de usuario, técnica en la cual se basa la presente investigación.

Finalmente, se puede decir que la Ingeniería de Usabilidad se puede definir como el proceso por el cual la usabilidad de un producto se especifica cuantitativamente. Después de que se construya el producto puede ser demostrado que alcanza, o no, los niveles requeridos de usabilidad. La Ingeniería de Usabilidad busca introducir medidas de usabilidad en todas las fases del desarrollo, además de conocer exactamente qué criterios se usarán para juzgar la usabilidad de un producto.

1.6 Test de usuario para el análisis de usabilidad

Existe una amplia variedad de métodos de evaluación de usabilidad (Ver Figura 2), cada uno de ellos utiliza determinados medios y técnicas e intentan medir diferentes aspectos. La elección de un método u otro no depende solo de cuál es la respuesta que se quiere conocer sino de múltiples factores que pueden resumirse en saber cuánto cuesta y qué se obtiene con su realización.

Figura 2. Clasificación de los métodos de evaluación. Elaborado según Griho[5].

Según la clasificación propuesta por Griho, en cuanto al **tipo de técnica** para la evaluación de la usabilidad se encuentra la técnica de test de usuario. En los métodos de evaluación por *test los* usuarios representativos trabajan en tareas concretas utilizando el sistema (o el prototipo) y los evaluadores utilizan los resultados para ver cómo la interfaz de usuario da soporte a los usuarios con sus tareas (GRIHO, 2010).

La evaluación de la usabilidad mediante la técnica de test de usuario abarca una serie de métodos que ayudan a medir la forma en que los usuarios son capaces de utilizar un producto, al mismo tiempo que determinan la manera en que lo hacen. El aplicar estos métodos contribuirá a la creación de mejores productos, donde los usuarios realicen sus actividades con mayor facilidad. Sin evaluación será imposible saber si un producto cumple las expectativas de los clientes a los que está destinado, o si se adapta a su contexto social, físico y organizativo.

[5]Grupo de Investigación en Interacción Persona Ordenador e Integración de Datos fundado por el Dr. Jesús Lorés

1.6.1 Test de usuario

Los test de usuario tienen la gran ventaja de que los resultados que se obtienen son fiables, pues son obtenidos directamente de los usuarios finales. No todos los test de usuario son iguales, por lo que, antes de empezar a trabajar con los usuarios, se debe planificar en cada caso cuales son las pruebas que van a formar del test. De forma genérica, las fases de un test de usuario son las siguientes:

Fases para la aplicación del *test* de usuario (CHILE., 2008) (Ver Figura 3).

1. Planificación: diseño del test y realización de la selección de los usuarios.
2. Ejecución: realización del test con los usuarios en el laboratorio o en el lugar de navegación de los mismos.
3. Conclusión: elaboración del informe final.

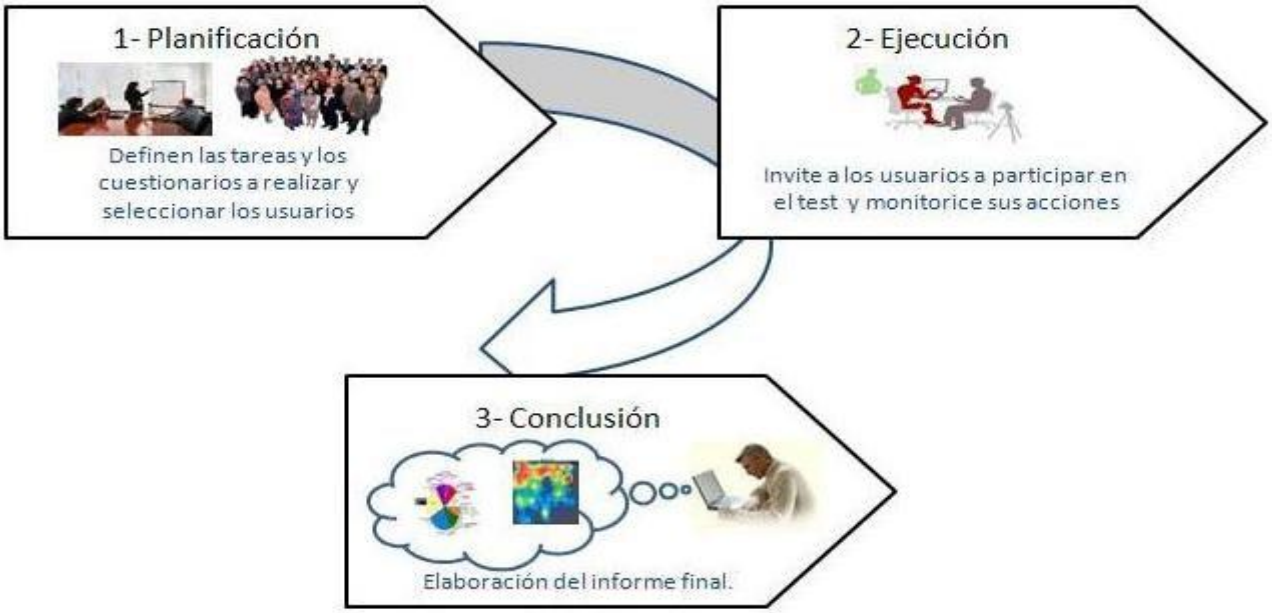


Figura 3. Fases para la aplicación del *test* de usuario.

A continuación se describen cada una de las fases:

Planificación: consiste en diseñar el test acorde con los indicadores seleccionados para la evaluación de la aplicación, los cuales son de vital importancia para los desarrolladores. El test se conforma elaborando las preguntas orientadas en que el usuario resuelva tareas. Estas permiten observar el grado de entendimiento que tiene la aplicación para cualquier tipo de usuario.

En el reclutamiento de participantes se debe asegurar de que los elegidos tienen perfiles acordes con los usuarios reales o potenciales del sistema (HASSAN, M. Y. S. HERRERO, V., 2007).Si son futuros usuarios del sistema el proceso es más efectivo.

Hassan y Martín describen que para realizar el test, se debe elegir los usuarios, con los que se pueden detectar hasta el ochenta por ciento de los problemas de usabilidad. En el mismo, se recomienda que la duración de los ejercicios no supere la hora, ya que el usuario tiende a agotarse, pierde el interés en lo que está haciendo y por ende, los resultados generados no serían los óptimos (HASSAN, Y. MARTÍN, F. J., 2003).

Ejecución: indica el lugar donde se debe aplicar el test de usuario. Se afirma que el mejor lugar es un laboratorio que cuente con la menor cantidad de personas posibles, para que no impidan la ejecución satisfactoria de las tareas. El evaluador debe ir anotando los problemas que va observando a medida que transcurra la ejecución del ejercicio. Las evaluaciones deben ser realizadas de forma independiente por cada usuario o participante (HASSAN, Y. MARTÍN, F. J., 2003).

Conclusión: se elabora un informe con las anotaciones tomadas durante la ejecución de las tareas. En el informe se deben incluir además de los problemas de usabilidad detectados, algunas sugerencias para solucionarlos. Algunos aspectos a tener en cuenta a la hora de realizar un test de usuarios es la preparación de los materiales (HASSAN, Y. MARTÍN, F. J., 2003).

1.6.2 Métodos de evaluación para la técnica test de usuario.

Medida de las prestaciones.

Este método de evaluación más conocido como test de usuarios está basado en la toma de medidas acerca del rendimiento u otro tipo de aspecto subjetivo que afecte a la usabilidad del sistema, para lo que será necesario disponer bien sea del sistema ya implementado o de un prototipo que permita evaluar estos aspectos. Se analizarán tanto en la manera como utilizan el producto como midiendo el tiempo que les lleva realizarlo (GRANOLLERS, 2004).

Pensando en voz alta

En este método de evaluación se pide a los usuarios y de forma individual que expresen en voz alta y libremente sus pensamientos, sentimientos y opiniones sobre cualquier aspecto (diseño, funcionalidad) mientras que interaccionan con el sistema o un prototipo del mismo. Resulta ser un método altamente eficaz para capturar aspectos relacionados con las actividades cognitivas de los usuarios potenciales del sistema evaluado (GRANOLLERS, 2004).

Interacción constructiva

Es una derivación del pensando en voz alta e implica tener, en vez de uno, a dos usuarios realizando conjuntamente cada test del sistema. Este método es mucho más natural que el pensar en voz alta con usuarios individuales, ya que las personas normalmente verbalizan cuando tratan de resolver un problema conjuntamente y además hacen muchos más comentarios (GRANOLLERS, 2004).

Test retrospectivo

Este método de test retrospectivo está basado en que posteriormente a la grabación del vídeo de la sesión de test, se recoge más información haciendo que el usuario revise la grabación. El evaluador detiene el vídeo y pregunta al usuario con más detalle del *test* que esencialmente ha sido completado (GRANOLLERS, 2004).

Método del conductor

El método del conductor trata de interferir lo menos posible al usuario mientras realizaba el test. Se conduce al usuario en la dirección correcta mientras se usa el sistema. Durante el test, el usuario puede preguntar al evaluador cualquier aspecto relacionado con el sistema y este le responderá. Este método se centra en el usuario inexperto y el propósito del mismo es descubrir las necesidades de información de los usuarios de tal manera que se proporcione un mejor entrenamiento y documentación, al mismo tiempo que un posible rediseño de la interfaz para evitar la necesidad de preguntas (GRANOLLERS, 2004).

Recoger los datos de clics

Esta técnica permite reconstruir las rutas de navegación de los usuarios así como dibujar "mapas de calor" sobre una página, mostrando los elementos con más clics. El mapa de clics muestra el lugar dónde hacen clics los usuarios (ROVIRA, 2011).

Seguidor de ojo

Está técnica hace referencia a un conjunto de tecnologías que permiten monitorizar y registrar la forma en la que una persona mira una determinada escena o imagen, en concreto en qué áreas fija su atención, durante cuánto tiempo y qué orden sigue en su exploración visual (GRAU, 2009).

Luego del estudio de las técnicas existentes para medir la usabilidad de un sistema se propone realizar el test de usuarios, apoyado en el método grabación de uso, como punto de partida para que el proyecto realice los restantes métodos de evaluación para llegar a una solución completa en la aplicación de este tipo de test. Por la importancia que tienen estas técnicas para el análisis del comportamiento de los usuarios en la web.

1.7 Herramientas homólogas

En muchos lugares del mundo, diferentes empresas han creado herramientas para evaluar la usabilidad de sus productos mediante la técnica de test de usuario. Lo cual ha posibilitado estudiar de forma precisa las reacciones de los usuarios, proporcionando de esta forma una interesante información para la toma de decisiones en materia de usabilidad y diseño de interfaces. A continuación se realiza un estudio de algunas de estas herramientas con el objetivo de conocer sus características, su funcionamiento y poder precisar elementos de interés para la elaboración de la herramienta a desarrollar.

UserZoom: es un sofisticado e innovador software de testeo remoto que permite analizar la usabilidad sobre grandes muestras de usuarios, geográficamente dispersos, de manera detallada, rápida y eficaz. Analiza información sobre las tareas más importantes que los usuarios realizan en una web, como el lugar exacto donde se han realizado los clics y el tiempo empleado. El usuario participa en el estudio desde su contexto natural, esto permite recoger opiniones y percepciones de forma real. Realiza las técnicas de: seguidor de ojo, test de usuarios, pensando en voz alta, mapa de clics y grabación de uso (USERZOOM, 2013).

ObRemUs: es un sistema software que permite la participación en pruebas de usabilidad remotas. Está compuesto por cuatro módulos independientes: módulo de grabación, módulo de moderación, módulo de evaluación y módulo de administración. Su objetivo es registrar los eventos de teclado y ratón, la actividad de la pantalla y la actividad del usuario. Ofrece al mismo el control total sobre el inicio y la finalización de las tareas planteadas. Permite visualizar la información registrada en un test de usuario y evaluar los resultados de este. Realiza las técnicas de: test de usuarios y grabación de uso (ALONSO, 2008).

Morae: es un paquete completo para realizar los exámenes. Permite identificar problemas de usabilidad en las aplicaciones de software, sitios web, prototipos, o dispositivos móviles. Además, permite la captura de cada matriz de la sesión de pruebas. Permite ver a los participantes y oír su opinión mientras descubren los problemas ocultos en el producto o en el sitio. Calcula automáticamente y grafica las métricas. Exporta gráficos, captura la pantalla y presenta vídeos en formatos estándar.Realiza las técnicas de: test de usuarios y grabación de uso. La herramienta consta de tres productos principales:

- *Recorder,* es el programa principal y permite grabar la sesión (pantalla, teclado, mouse) y al usuario al mismo tiempo.
- *Observer*, se utiliza para ver en tiempo real y de forma remota el *test* de usabilidad y permite realizar notas sobre el mismo.
- *Manager,* sirve para tomar decisiones en base a los resultados obtenidos en cada examen, permite analizar las grabaciones y generar reportes (CARVAJAL y SAAB, 2010).

Usertesting: es una herramienta bajo licencias propietarias. Para la configuración de un test se especifica

la dirección del sitio y se describe el escenario que se le plantea al participante. El análisis de los resultados cuenta con un video de grabación del rastro del ratón durante la sesión de *test* acompañado por la grabación de la voz de los usuarios. También un resumen de sugerencias que los usuarios proponen para cada sitio web y tarea. Es una herramienta que permite realizar: test de usuarios, hablando en voz alta, mapa de clics y grabación de uso (RODRÍGUEZ, 2009).

Loop11: es una herramienta web, se configura rápido y fácil. Permite la realización de test de usuarios con un máximo de cinco tareas a desarrollar y dos preguntas. Cuenta con varios idiomas para la interfaz, estos son inglés, español, alemán y chino. Al concluir el test de usuario proporciona datos que incluyen la tasa media de éxito y de fracaso, y el abandono del test completo. Sobre cada una de las tareas ofrece indicadores como: promedio de páginas vistas y tiempo medio por tarea, página más frecuente de éxito, fracaso y abandono, el primer clic y la ruta de navegación más frecuente. Además, de los indicadores anteriores, recoge una gran cantidad de datos sobre cada participante, tales como la dirección IP, navegador, fecha del test, tiempo total y por tarea del test, número total y por tarea de las páginas vistas. Trabaja bajo licencias libres. Realiza las técnicas de: test de usuarios y grabación de uso (LOOP11, 2013).

Silverback: es una herramienta bajo licencias propietarias. Permite realizar test de usabilidad y grabar toda la actividad de la pantalla mientras el usuario hace uso de una aplicación a la vez que graba su cara y su voz para registrar las diferentes reacciones. También permite añadir indicadores que posibilitan crear marcas para delimitar tareas o los problemas de usabilidad que se encuentren. Por último, da la posibilidad de exportar los vídeos para su visualización. Su gran deficiencia es que no se pueden medir las principales métricas de usabilidad y no existe alguna funcionalidad básica para la explotación de estos datos. Es una aplicación ideal para aquellos que no quieran realizar inversiones más costosas, ya que la licencia solo cuesta 49,95 dólares. Realiza las técnicas de: test de usuarios y grabación de uso (SILVERBACK, 2010).

AttentionWizard: creado por los expertos de los SiteTuners.com, utiliza un algoritmo avanzado para simular "mapas de calor", estas imágenes ayudarán a tener una idea de cuáles son los elementos de su sitio que atraen mayormente la atención, así como el orden en que las personas son atraídos a ellos. Se encuentra actualmente en una versión beta, así que el acceso a ella es un poco limitado. Solo está disponible como un informe PDF. Trabaja bajo licencias libres. Realiza la técnica de: mapa de clics (ATTENTIONWIZARD, 2013).

ClickDensity: es una herramienta bajo licencias propietarias. Permite mediante el seguimiento de los clics de los visitantes producir un mapa que muestra los clics como puntos calientes de actividad. Presenta opciones de personalización que incluyen el filtrado prototipo de navegador y rango de fechas, así como un ajuste de transparencia útil para ver el mapa de calor con mayor claridad. Realiza la técnica de: mapa de clics (CLICKDENSITY, 2011).

CrazyEgg: es una herramienta bajo licencias propietarias. Genera una variedad de visualizaciones basadas en la actividad de los usuarios en la pantalla a través de los clics, lo cual genera un mapa de calor tradicional que permite el seguimiento de clics por fuente y otras métricas de visitantes. Es muy fácil de usar y las opciones de personalización son limitadas. Realiza la técnica de: mapa de clics (CRAZYEGG, 2008-2013).

Chalkmark: es una herramienta bajo licencias propietarias. Permite cargar una serie de imágenes y crear tareas. Permite a los usuarios participar a través de un enlace personalizado para responder diferentes tareas. Posibilita guardar los clics de grabación para medir la realización de tareas. Los resultados de esta prueba de usuario simplificada se presentan como un mapa de calor que indica donde los usuarios hacen clic para completar cada paso en la encuesta. Realiza la técnica de: mapa de clics, test de usuario y grabación de uso (CHALKMARK, 2012).

A continuación se muestra una tabla resumen con las características que engloban cada una de las herramientas evaluadas y en donde se realiza una comparación entre ellas. Para definir aspectos que pudieran ser incluidos en la solución propuesta en esta investigación. Los elementos a comparar son los siguientes:

1. Realización de las técnicas para medir la usabilidad, dígase test de usuario, pensando en voz alta, grabación de uso, mapa de clics, encuesta y seguidor de ojo.
2. Si son privativas o no privativas, lo cual permite determinar si se puede obtener sin la necesidad de comprar una licencia o contratar a consultorías para obtener su servicio.

Características / **Herramientas**	**Test de usuario**	**Pensando en voz alta**	**Grabación de uso**	**Mapa de clics**	**Encuestas**	**Seguidor de ojo**	**No privativas**
UserZoom	x	x	x	x	x	x	
Morae	x		x				

ObRemUs	x		x				
UserTesting	x	x	x	x			
Loop11	x		x				x
Silverback	x		x				
AttentionWizard				x			x
ClickDensity				x			
Chalkmark	x		x	x			
CrazyEgg				x			

Tabla 1 Comparación entre herramientas homólogas.

Cabe destacar que la herramienta más completa es **UserZoom** pero es una herramienta que trabaja bajo licencias propietarias, por lo que se dificulta su utilización. Mientras que las herramientas que trabajan bajo licencias libres solo realizan algunas de estas técnicas, como **AttentionWizard**, que solo realiza la técnica mapa de clics y **Loop11**, que realiza test de usuarios y grabación de uso, además de solo permitir la realización de cinco tareas y por esta razón se dificulta su utilización. Por ello sería de mayor utilidad para los especialistas tener una herramienta más completa en la cual estén presentes todas las técnicas mencionadas en la tabla comparativa, excepto seguidor de ojo. Esta última técnica se considera interesante debido a que muchas empresas a nivel internacional como Google, Facebook (DIOSDADO, 2012), Microsoft (ALTOFT, 2009), entre otras la utilizan para analizar el comportamiento que tiene los usuarios frente a la información que reciben de los diferentes sitios y portales; pero por cuestiones tecnológicas y económicas se debe prescindir de su utilización. Este trabajo de diploma sentará las bases

de la herramienta en la implementación de la técnica de test de usuario apoyada en el método de grabación de uso, como punto de partida para que el proyecto realice los restantes métodos de evaluación para llegar a una solución completa en la aplicación de este tipo de test.

1.8 Tecnologías para el desarrollo de la propuesta

1.8.1 Metodología de desarrollo

Las metodologías de desarrollo de software indican cómo hay que obtener los distintos productos parciales y finales. Definen un conjunto de procesos y actividades que guían a los desarrolladores 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 a utilizar en este caso son XP para el desarrollo del sistema y Scrum para la gestión de proyecto.

XP (Programación Extrema, por su traducción al español)**:** es una metodología ágil de desarrollo de software que posee cuatro tareas fundamentales: planificación, diseño, desarrollo y pruebas. Esta metodología está basada en la simplicidad durante el desarrollo, la comunicación entre las partes implicadas (clientes y desarrolladores) y la retroalimentación para poder reutilizar el código desarrollado. En su concepción establece entregas frecuentes con posibilidad de refactorización continua, permitiendo mejorar el diseño cada vez que se añade una funcionalidad. Para su implementación XP establece un conjunto de prácticas que deben ser empleadas en los proyectos de desarrollo, las mismas se mencionan a continuación:

- El juego de la planificación
- Entregas pequeñas
- Metáfora
- Diseño simple
- Pruebas
- Refactorización
- Programación en parejas
- Propiedad colectiva del código
- Integración continua
- 40 horas por semana
- Cliente in-situ
- Estándares de programación (LETELIER; SINHA, 2010) .

Scrum: es una metodología ágil enfocada a la gestión de proyectos. Sus principales características se pueden resumir en dos: el desarrollo de *sprint o* iteraciones y reuniones a lo largo del desarrollo. Las iteraciones en Scrum tienen una duración máxima de 30 días y el resultado de cada una de ellas define un

incremento del producto a desarrollar. La evolución del proyecto por la metodología se define a través de reuniones diarias donde el trabajo del día anterior es revisado por el equipo, previendo además la labor a realizar el día siguiente. Dentro de las prácticas definidas por la metodología Scrum se encuentran:

- Planificación de la iteración o *sprint*
- Revisión de la iteración o *sprint*
- Reunión diaria
- Pila del producto
- Incremento
- Propietario del producto
- Auto-organización (PALACIO, 2008).

1.8.2 Lenguaje de modelado

Lenguaje unificado de modelado (UML) 2.0: es el lenguaje que se usa para visualizar, especificar, construir y documentar un sistema, de forma gráfica, característica que ayuda mucho en la comprensión de los sistemas. UML también intenta solucionar el problema de diversidad de código que en ocasiones se presenta entre los desarrolladores, pues al implementar un lenguaje de modelado común para todos los desarrollos, se crea una documentación también común, que cualquier desarrollador con conocimientos de este lenguaje será capaz de entender. Se propone utilizar este lenguaje pues permite modelar sucesos de la vida real con mucha facilidad y además brinda la posibilidad de adaptar el sistema a cualquier lenguaje de programación, factor que lo hace reutilizable (PRESSMAN, 2002).

1.8.3 Herramienta de modelado

Visual Paradigm 8.0: es una herramienta pensada para documentar elementos del ciclo de vida de un proyecto. Permite elaborar todos los diagramas de clases, permite además realizar código inverso, es decir, generar código desde diagramas y generar documentación. Tiene la capacidad de integrarse a entornos de desarrollo como el Netbeans. Es multiplataforma y por lo tanto puede correr tanto sobre plataformas libres como sobre plataformas privativas. También presenta como una ventaja fundamental que a través de este se pueden realizar prototipos de interfaz de usuarios que permiten tener una visión más cercana de cómo quedarían las interfaces del sistema.

1.8.4 Lenguaje de programación

Java: permite que los desarrolladores creen un sistema informático en una plataforma y ejecutarlo en otra distinta de la que se creó inicialmente. Se caracteriza por su potencia, pero a la vez elimina las características menos usadas y más complejas de otros lenguajes como C y C++. Con este lenguaje se crean programas tanto para navegadores web, servicios web, aplicaciones potentes y eficientes para teléfonos móviles, procesadores remotos, productos de consumo de bajo costo y para cualquier tipo de

dispositivo digital (GARCÍA, 2009). Se selecciona este lenguaje de programación pues es el definido por el proyecto.

1.8.5 Ambiente de Desarrollo Integrado (IDE)

Netbeans 7.2.1: Es un proyecto de código abierto, gratuito y sin restricciones de uso, que permite la extensión de aplicaciones hechas en este entorno dada su característica de que los módulos pueden ser desarrollados de manera independiente. En muchas ocasiones los diseñadores buscan un ambiente gráfico agradable para sus aplicaciones y el Netbeans ofrece en su biblioteca AWT/Swing para gestionar interfaces de usuarios (SILVA, 2009).

Conclusiones parciales

En el presente capítulo se analizaron los conceptos fundamentales que guían la investigación, lo cual permitió lograr una vista panorámica general de la usabilidad, así como de comprender por qué es necesaria. Se realizó además un análisis de soluciones existentes con características similares a la propuesta solución lo cual contribuyó a determinar la necesidad de implementar una solución que apoye la toma de decisiones de los profesionales en el proceso de desarrollo de interfaces de usuario. Además, se realizó la selección de las herramientas, tecnologías y la metodología para el desarrollo del sistema, lo cual permitió darle solución a la problemática planteada, teniendo en cuenta las características de la presente investigación y partiendo del concepto de migración a software libre que sigue el país actualmente.

Capítulo 2: Implementación de la solución propuesta

Introducción

En el presente capítulo se realiza una valoración crítica del análisis y se caracteriza el sistema propuesto por la analista. Se describe el diseño de la propuesta solución. Se realiza la selección de los patrones arquitectónicos y de diseño a utilizar. Se define el estándar de codificación a utilizar. Se realiza la modelación del sistema, a través del diagrama de paquetes, entre otros.

2.1 Valoración crítica del análisis

En análisis y diseño del sistema propuesto en el Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas titulado *"Propuesta de herramienta para análisis de usabilidad en el paquete Abad" (LORENZO ESCOBAR, 2012)*, se decidió desarrollar una herramienta para realizar análisis heurístico y test de usuarios. Sin embargo, luego de una revisión de dicho análisis se llegó a la conclusión de no incluir el módulo de análisis heurístico y del módulo test de usuario, los métodos de evaluación hablando en voz alta, mapa de clic y encuesta. Lo cual no significa que se excluya en versiones posteriores. Se determinó además realizar algunos cambios en el diseño de interfaz debido a que la propuesta de diseño existente no cumple con las especificaciones necesarias para el cumplimiento de cada requisito. Para finalizar se realizó un examen crítico de las funcionalidades necesarias concluyendo que solo serían obviadas las funcionalidades que respectan al análisis heurístico y las referentes a los métodos de evaluación hablando en voz alta, mapa de clic y encuesta por parte de la aplicación. Luego de realizada todas estas precisiones se puede concluir que la herramienta a desarrollar permitirá la ejecución de la técnica de test de usuario, apoyada por el método de evaluación grabación de uso.

2.2 Propuesta de solución

A continuación se presenta la concepción de la herramienta elaborada por la analista, la cual reúne un primer acercamiento las características básicas para cubrir las necesidades que puedan tener los evaluadores expertos en usabilidad cuando realizan la evaluación.

La herramienta consta de dos aplicaciones, una para el trabajo del especialista (aplicación máster) y otra para interactuar con el usuario (aplicación cliente).

La aplicación para el trabajo del especialista presenta los siguientes módulos:

- Módulo general.

- Módulo de test de usuarios.

En los siguientes apartados se detallan las características principales de los módulos mencionados, ya que en esta primera versión serán los únicos a incluir.

2.2.1 Módulo general

El módulo general es la parte de la herramienta que almacenará toda la información necesaria para poder realizar el análisis de usabilidad de un sistema interactivo. Esta información puede ser la requerida por los proyectos y/o generada en las máquinas clientes.

Desde este módulo se pueden visualizar durante el tiempo de ejecución del ejercicio, las pantallas de las máquinas clientes. Esto permitirá monitorizar el comportamiento de los clientes durante la prueba, analizando su comportamiento al interactuar con el sitio a evaluar. Permite además guardar y cargar las tareas, así como los resultados obtenidos en el test realizado, para en caso de que se desee realizar nuevamente el mismo o consultar sus datos.

2.2.2 Módulo de test de usuario

En este módulo el especialista inicia haciendo los preparativos, define las tareas a realizar por los usuarios y selecciona las técnicas a desarrollar. Posteriormente le hace llegar el test preparado a los usuarios, los cuales reciben una notificación que informa dónde acceder para abrir la aplicación a probar. En el desarrollo del test por los usuarios, se recogen un conjunto de elementos que son la base para los resultados finales, los mismos se describen a continuación:

- Grabar la actividad en pantalla, para posteriormente obtener un video que permita estudiar el comportamiento del usuario interactuando con la interfaz.
- Determinar las tareas completadas por los participantes para definir si los usuarios acaban las tareas con éxito.
- Recopilar el tiempo de realización de cada tarea por usuario, para junto al número de clics obtener cuánto tiempo y esfuerzo dedican a completar dichas tareas (LORENZO ESCOBAR, 2012).

La información se envía al módulo general, todos estos elementos se procesan para adquirir los resultados, tiempo de realización de una tarea medio, máximo y mínimo. Así como el porciento de tareas completadas y no completadas por los usuarios. Todo esto se muestra de forma gráfica al especialista en una interfaz. Además, se envía al módulo general la grabación en pantalla que permita estudiar el comportamiento del usuario al interactuar con la aplicación a evaluar.

En la aplicación para interactuar con el usuario, primeramente se establece la conexión con la aplicación máster. Posteriormente recibe una notificación bienvenida que informa la dirección del sitio a evaluar. Luego recibe el test preparado por el especialista, donde tendrá las opciones aceptar y terminar las tareas

recibidas o abandonarlas. Al concluir con todas las tareas recibidas se muestra una notificación donde se informa que ha concluido el test.

2.3 Patrón arquitectónico

En la tecnología de objetos un patrón es una descripción de un problema y la solución, a la que se le da nombre, y que se puede aplicar a nuevos contextos. En el caso de los patrones arquitectónicos proponen un esquema organizativo estructural para los sistemas informáticos (KRUCHTEN, 1995) .

2.3.1 Patrón arquitectónico en tres capas

Una arquitectura de software diseñada en capas consiste en la definición de niveles de abstracción, los cuales tienen una función específica permitiendo un diseño modular. La programación por capas es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. Una variante de este patrón muy utilizada es la de tres capas. Permite dividir la aplicación informática en tres capas, capa de presentación, capa de lógica de negocio y la capa de acceso a datos.

- **Capa de presentación**: es la que se observa, la que comunica la información y captura la información del usuario. Esta capa se comunica únicamente con la capa de negocio.
- **Capa de negocio**: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario. Es donde se establece todas las reglas que se deben cumplir. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él.
- **Capa de acceso a datos**: es la que gestiona el almacenamiento de los datos, y 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).

Se adoptó la utilización del patrón arquitectónico de tres capas, debido que permite estructurar mucho más las aplicaciones que se desarrolla, además de depurar más fácilmente los errores que van surgiendo a medida que la aplicación va evolucionando. Es factible y cómodo para desarrollar varias personas al mismo tiempo dentro de un equipo, proporcionando una distribución clara del trabajo entre los miembros de un equipo de desarrollo. Proporciona una buena organización y reutilización del código generado. Además, es la arquitectura que se adoptó para el desarrollo de la solución general.

2.4 Patrones de diseño

Un patrón de diseño es una solución estándar para un problema común de programación, siendo más práctico a la hora de describir ciertos aspectos de la organización de un programa y conexiones entre componentes de programas.

2.4.1 Patrones Generales de Software para la Asignación de Responsabilidades (GRASP)

Los patrones GRASP (*General Responsibility Assignment Software Patterns*, en español Patrones Generales de Software para la Asignación de Responsabilidades) describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable (KUCHANA, 2004).

En el desarrollo de esta solución se aplicaron los patrones GRASP: Experto, Controlador, Bajo Acoplamiento, Alta Cohesión y Creador.

- **El patrón Experto**: soluciona el problema de asignar una responsabilidad de forma general, tomando decisiones sobre la asignación de responsabilidades a las clases. Si se hacen en forma adecuada, los sistemas tienden a ser más fáciles de entender, mantener y ampliar.
- **El patrón Controlador**: asigna la responsabilidad de recibir o manejar los mensajes de evento del sistema.
- **El patrón Bajo Acoplamiento**: da soporte a una dependencia escasa y a un aumento de la reutilización, manteniendo las clases que no dependan de muchas otras clases.
- **El patrón Alta Cohesión**: aconseja mantener la clase lo más cohesionada posible para controlar la complejidad de la misma.
- **El patrón Creador**: asignar a la clase B la responsabilidad de crear una instancia de clase A si se cumple uno o más de los casos siguientes:

  B agrega objetos de A.

  B contiene objetos de A.

  B registra instancias de objetos de A.

  B utiliza más estrechamente objetos de A.

  B tiene los datos de inicialización que se pasarán a un objeto de A cuando sea creado (por tanto, B es un experto con respecto a la creación de A).

  B es un creador de los objetos A.

  Si se puede aplicar más de una opción, inclínese por una clase B que agregue o contenga la clase A.

2.4.2 Patrones GoF

Son patrones de diseño creado por “La Pandilla de los Cuatro” (*Gang of Four, GoF*). Se utilizarán los patrones Fábrica y Fachada.

- **Fábrica:** su propósito es crear objetos, permitiendo al sistema identificar que clase se debe instanciar en tiempo de ejecución.
- **Fachada:** proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace que el subsistema sea más fácil de usar (LARMAN, 1999).

2.5 Estándar de codificación

Es prudente establecer estándares de codificación para todos los programadores. Estos estándares consisten en estilos de codificación a la hora de escribir el código. Mantener un estándar en el código es muy recomendable, ayuda a mantener proyectos ordenados y de fácil lectura para el resto de las personas. Luego, es una buena práctica que sigas siempre los consejos de un buen estándar.

La legibilidad del código fuente repercute directamente en el entendimiento que pueda tener otro programador del mismo, aspecto crucial ya que todo software tiene que someterse constantemente a mantenimiento y mejora de sus funcionalidades. El mejor método para lograr que un grupo de desarrolladores mantenga un código de calidad es establecer un estándar de codificación sobre el cual se realizarán revisiones rutinarias (ESTÁNDAR_DE_CÓDIGO, 2010).

Existen diferentes estándares de código, uno de ellos es la notación Camel, esta consiste en escribir los identificadores con la primera letra de cada palabra en mayúsculas y el resto en minúscula. Se llama notación Camel porque los identificadores recuerdan las jorobas de un camello. Existen dos variantes:

- UpperCamelCase, CamelCase o PascalCase: en esta variante la primera letra también es mayúscula.
- LowerCamelCase, camelCase o dromedaryCase: la primera letra es minúscula.

Al conformar un estándar de codificación deben tenerse en cuenta varios criterios entre los cuales se pueden mencionar:

- **Nombres representativos para clases, variables y procedimientos***:* se definen nombres de variables, controles y procedimientos de forma representativa con el propósito de facilitar el entendimiento.
- **Indentación (sangrías) y espacios apropiados en el código:** visualizar el código fuente puede resultar complicado pero haciendo uso de la indentación se obtiene una mejor visibilidad pues se muestran las líneas que están sujetas a otras.

- **Documentación del código (poner comentarios para aclarar):** es una buena práctica documentar aquellas secciones de código más complicadas e inusuales para que pueda ser comprendida posteriormente.
- **Procedimientos coherentes:** no debe añadírsele demasiada complejidad a los procedimientos para que sean más fáciles de entender y no tengan una alta probabilidad de ocurrencia de errores.

Se especifican a continuación los criterios de estandarización de código utilizados para el desarrollo de la aplicación.

2.5.1 Nomenclatura de las clases, variables y procedimientos

Para las clases se establece UpperCamelCase que define que la primera letra de cada una de las palabras es mayúscula con la peculiaridad que las clases poseen el prefijo **U**_.

```
/**
 *
 * @author jose
 */
public class U_ClientePrincipal {
```

Figura 4: Ejemplo de nomenclatura UpperCamelCase

Para las variables se establece lowerCamelCase donde es igual que la anterior con la excepción de que la primera letra es minúscula y su nombre debe guardar relación con el valor que almacena.

```
private long tiempoInicio;
private long tiempoFin;
private boolean isAceptada;
private LinkedList<U_PuntoClick> listadoClick;
```

Figura 5: Ejemplo de nomenclatura lowerCamelCase

2.5.2 Indentación

Usar una indentación sin tabulaciones, con un equivalente a cuatro espacios, para mantener la organización y uniformidad del código. La longitud de las líneas de código es aproximadamente de 75 a 80 caracteres para mantener la legibilidad del código. Cuando una línea no quepa en una única línea se debe fraccionar atendiendo a estos principios generales:

- Fraccionar después de una coma.
- Fraccionar después de un operador.

```
Object object = in.readObject();
if (object instanceof U_Tarea_Objeto &&
    !((U_Tarea_Objeto) object).isVieneCliente()) {
        U_Tarea_Objeto lis = (U_Tarea_Objeto) object;
        testCliente.setListadoTarea(lis.getListadoTarea());
        testCliente.getVista();
}
```

Figura 6: Código Indentado.

2.5.3 Estructura de control.

Entre la estructura de control (*if, for, foreach, while, switch*), y los paréntesis deben de existir siempre un espacio. Se recomienda utilizar siempre llaves de apertura y cierre. Esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos.

```
public int cantidadTareaAceptada() {
    int cantidad = 0;
    for (int i = 0; i < listadoTareasResueltas.size(); i++) {
        if (listadoTareasResueltas.get(i).isIsAceptada()) {
            cantidad++;
        }
    }
    return cantidad;
}
```

Figura 7: Estructura de control.

2.5.4 XML (Lenguaje de Etiquetado Extensible).

Es un lenguaje muy simple, pero estricto que desempeña un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML, pero su función principal es describir datos y no mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones.

Los esquemas XML definen los elementos y atributos presentes en un documento y son la base de la

organización de la información del mismo (MERCER, 2001). La solución debe recuperar información de ficheros XML, por lo que es necesario un esquema para garantizar la organización de la información en el archivo. A continuación se expone un ejemplo de código XML:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <TestUsabilidad>
  - <Proyecto nombre="Biblioteca" URL="http://biblioteca.uci.cu">
      <Tecnicas grabar="true" hablar="false" encuesta="false" mapaclic="false" />
    - <Tareas>
        <Tarea nombre="Tarea#1" descripcion="Reservar un libro" />
        <Tarea nombre="Tarea#2" descripcion="Buscar un libro por su autor" />
        <Tarea nombre="Tarea#3" descripcion="Buscar otras bibliotecas" />
        <Tarea nombre="Tarea#4" descripcion="Regresar al home" />
      </Tareas>
    - <TareasResueltas>
        <TareaResuelta nombre="Tarea#4" descripcion="Regresar al home" tiempoInicio="1368456046900"
          tiempoFin="1368456059434" aceptada="true" />
        <TareaResuelta nombre="Tarea#3" descripcion="Buscar otras bibliotecas" tiempoInicio="1368456015644"
          tiempoFin="1368456017608" aceptada="true" />
        <TareaResuelta nombre="Tarea#2" descripcion="Buscar un libro por su autor" tiempoInicio="1368455972924"
          tiempoFin="1368455982727" aceptada="true" />
        <TareaResuelta nombre="Tarea#1" descripcion="Reservar un libro" tiempoInicio="0" tiempoFin="0" aceptada="false" />
      </TareasResueltas>
    </Proyecto>
</TestUsabilidad>
```

Figura 8: Ejemplo de código XML.

2.6 Modelo de diseño

Para una mejor comprensión y organización de la solución teniendo en cuenta la utilización del patrón arquitectónico de tres capas se han organizado las clases en tres paquetes principales: Presentación, Lógica el negocio y Acceso a datos. Cada paquete se ha dividido a la vez en subpaquetes de acuerdo con las funcionalidades de las clases contenidas. Cada paquete representa una capa de la arquitectura y está construido sobre la capa que la precede. 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 (Ver Figura 10).

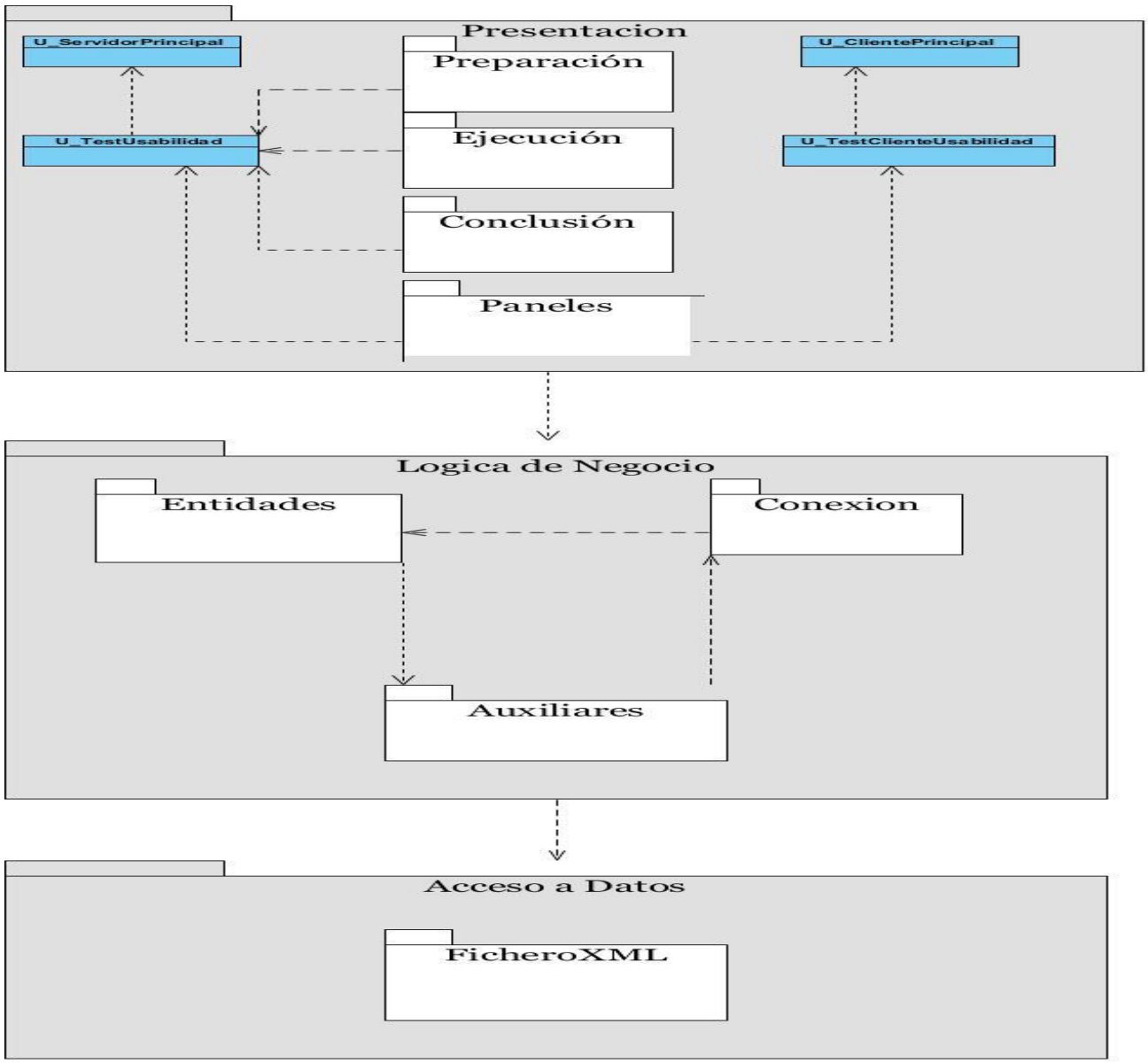


Figura 9: Diagrama de paquetes del sistema

2.6.1 Capa Presentación

La capa de Presentación es la encargada de lograr la comunicación directa entre el sistema y el usuario final a través de una interfaz gráfica. La interfaz gráfica de usuario consiste en un conjunto de componentes empleados por los 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 son Preparación, Ejecución, Conclusión y Paneles (Ver Figura 10). A continuación se explicará en que consiste cada uno de estos paquetes.

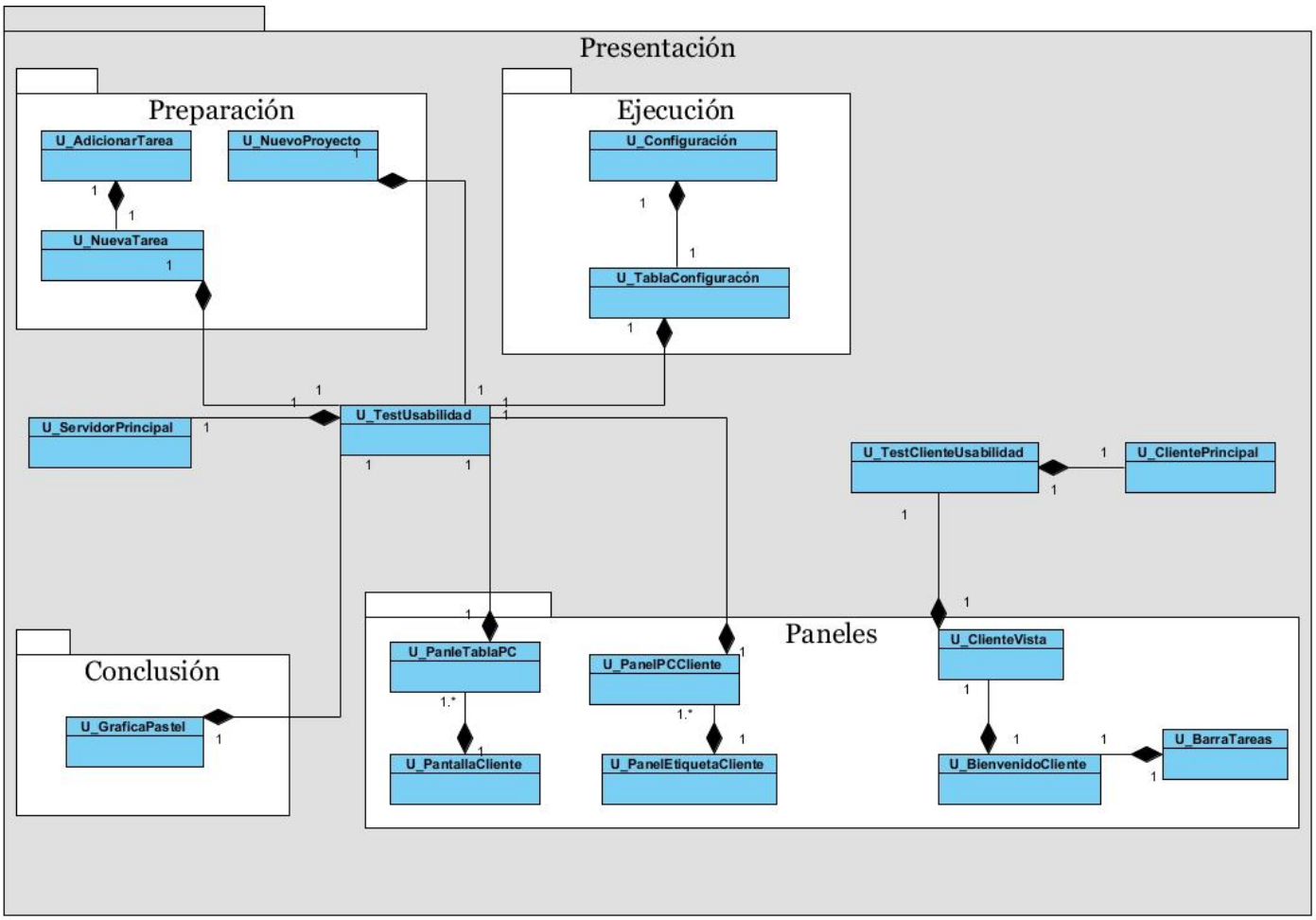


Figura 10: Capa presentación

En el paquete Preparación se pueden encontrar las interfaces para visualizar la gestión de las tareas, en donde el especialista podrá realizar una serie de acciones como adicionar las tareas del test, listarlas, eliminarlas, editarlas y cambiarles el orden de aparición. Además, desde en este paquete también se puede acceder a la interfaz para crear proyectos, permitiendo definir un nombre para el proyecto, indicar la dirección del sitio a evaluar y seleccionar los métodos de evaluación a utilizar en cada caso (Ver Anexo 1).

En el paquete Ejecución se pueden encontrar las interfaces para visualizar las opciones de configurar las máquinas clientes que intervendrán en la realización de las pruebas; en estas interfaces se podrá listar, adicionar, modificar y eliminar cada una de las máquinas (Ver Anexo 2).

En el paquete Conclusión se pueden encontrar las interfaces para visualizar los resultados obtenidos luego de la realización del ejercicio, como son: el porciento de tareas abandonadas y aceptadas, tiempo medio, mínimo y máximo de realización de las tareas. Además, permite la visualización de las grabaciones de actividad de los usuarios durante la ejecución de la prueba (Ver Anexo 3).

En el paquete Paneles se pueden encontrar las interfaces para visualizar el panel lateral en donde se encuentran listadas cada una de las máquinas clientes utilizadas en la realización del ejercicio. Además,

un panel para el área de trabajo en donde el especialista podrá monitorizar la actividad del cliente durante la realización de la prueba. Se pueden encontrar también las interfaces que permiten visualizar el mensaje de bienvenida en las máquinas clientes, así como las tareas que han sido creadas para la realización del ejercicio (Ver Anexo 4).

2.6.2 Capa lógica del negocio.

Es donde se establece todas las reglas que se deben cumplir. Esta capa se comunica con la capa de Presentación, para recibir las solicitudes y con la capa de datos para solicitar almacenar los datos en ficheros o en archivo XML o recuperar datos de él. En el diseño orientado a objetos la capa lógica del negocio se divide en otras menos densas la cuales son Entidades, Conexión y Auxiliares (Ver Figura 11). A continuación se explicará en que consiste cada uno de estos paquetes.

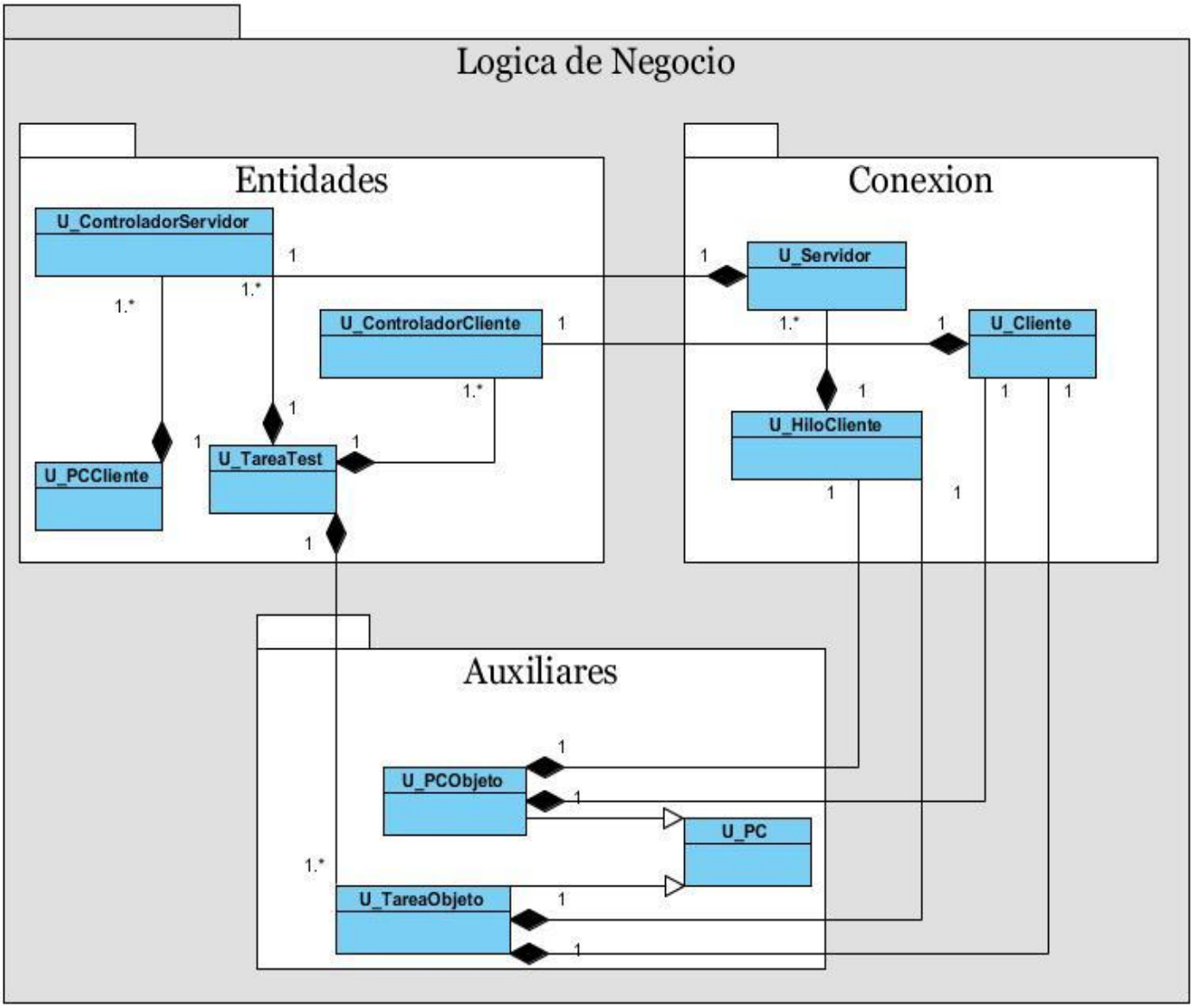


Figura 11: Capa de lógica del negocio

En el paquete Entidades se encuentran las clases **U_TareaTest**, en la cual se crea el objeto tareas y se definen los atributos que la caracterizarán como: nombre, descripción, si es aceptada, tiempo de inicio y tiempo de fin; **U_PCCliente**, en la cual se crea el objeto correspondiente a las máquinas clientes y se definen los atributos que la caracterizarán como: nombre y dirección IP; **U_ControladorCliente**, la cual es la encargada de controlar el negocio en la parte de la aplicación cliente. En ella se puede definir el listado de tareas que serán realizadas por los clientes. Finalmente, se encuentra la clase **U_ControladorServidor**, la cual es la encargada de controlar el negocio en la parte de la aplicación máster. En ella se puede definir el listado de tareas que serán realizadas por los usuarios y un listado de máquinas clientes.

En el paquete Conexión se encuentran las clases **U_Cliente**, la cual es la encargada de conectar la máquina cliente con la máquina máster, de enviar y recibir los datos; **U_Servidor**, que es la encargada de crear la conexión para cada PC cliente, estableciendo de esta forma un hilo de conexión. Finalmente, se encuentra la clase **U_HiloCliente**, la cual es la encargada de enviar y recibir los datos.

En el paquete Auxiliares se encuentran las clases que crean los objetos que se enviarán y recibirán en cada caso.

2.6.3 Capa de acceso a datos

La capa de acceso a datos es la encargada de la persistencia y recuperación de objetos. 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 (Ver Figura 12). A continuación se explicará en que consiste cada uno de estos paquetes.

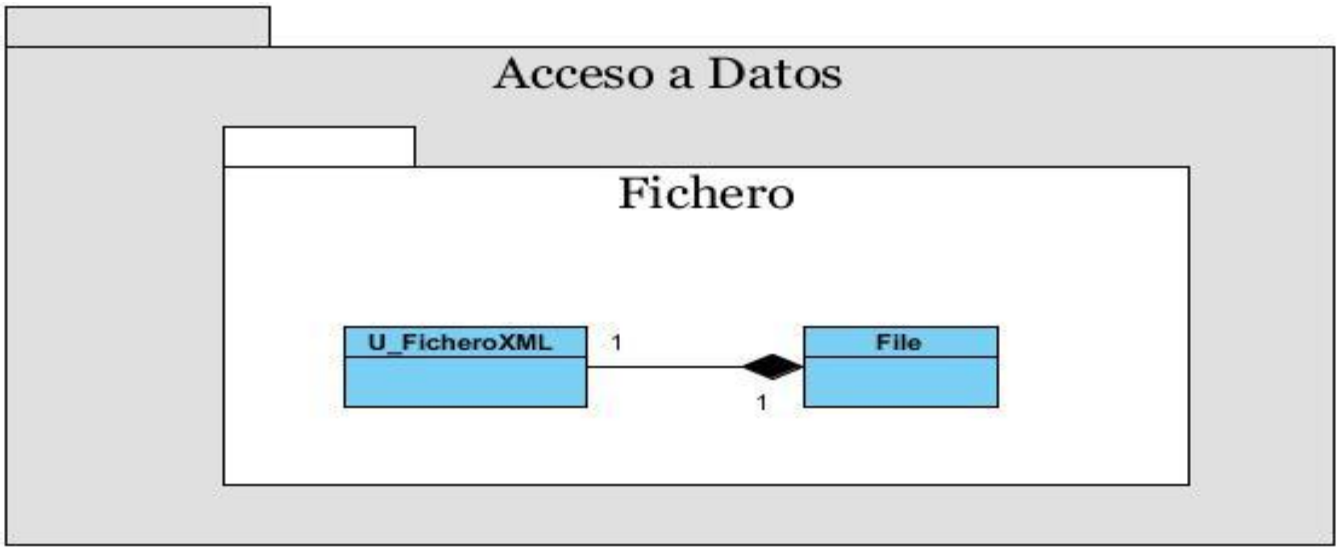


Figura 12: Capa de acceso a datos

En el paquete Fichero se encuentran las clases encargadas de la persistencia de los datos. Permite guardar en un fichero XML los datos, así como cargar los mismos.

2.7 Modelo de implementación

El modelo de implementación describe cómo los elementos del modelo de diseño y las clases, se implementan en términos de componentes, como ficheros de código fuente, ejecutables, entre otros. El modelo de implementación describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y cómo dependen unos de otros (JACOBSON, 2000). A continuación se muestra el modelo de implementación de la solución.

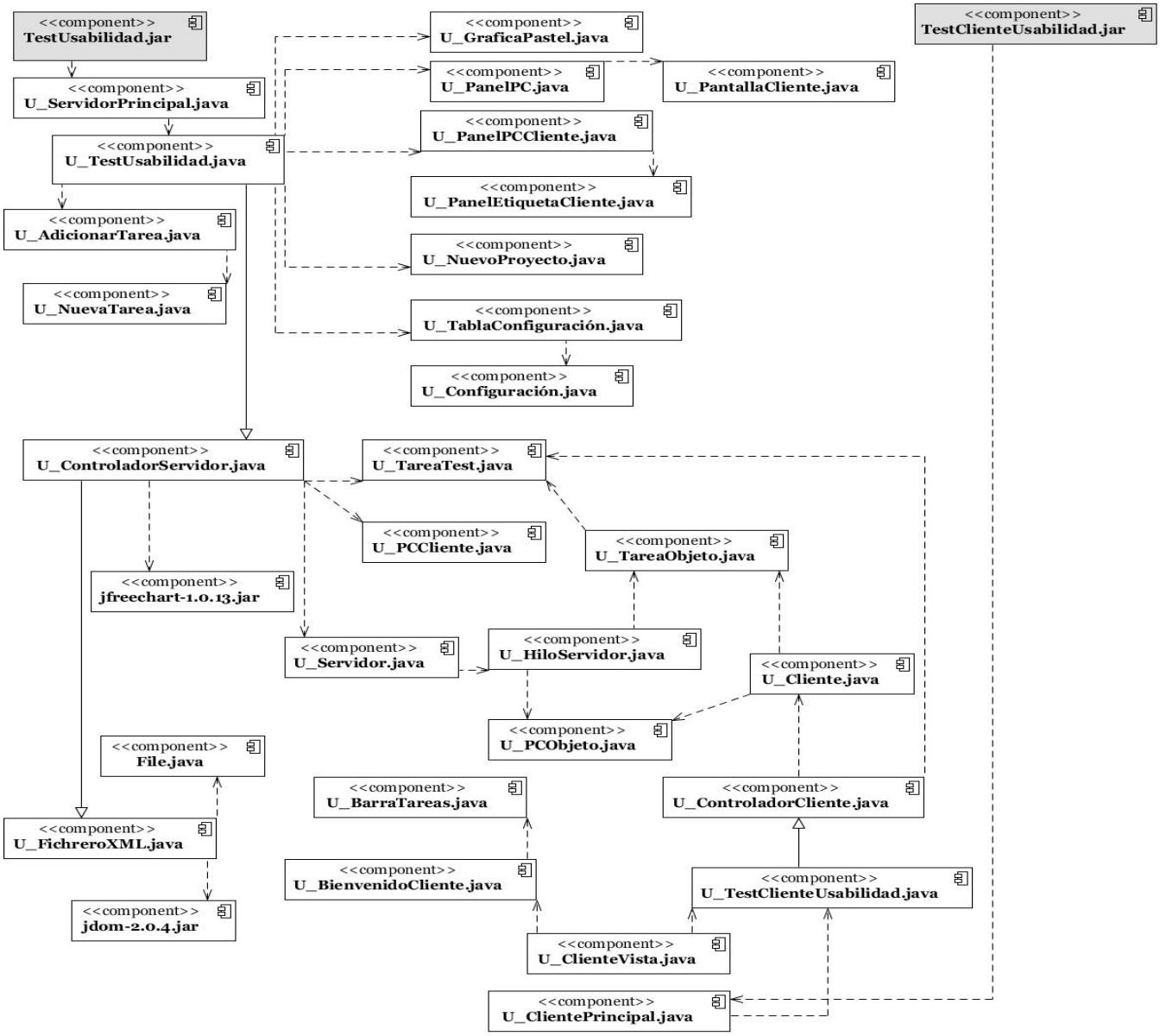


Figura 13: Modelo de implementación

2.8 Modelo de despliegue

Es un modelo que muestra la disposición de las particiones físicas del sistema de información y la asignación de los componentes software a estas particiones. Es decir, las relaciones físicas entre los componentes software y hardware en el sistema a entregar (JACOBSON, 2000). La presente solución es una aplicación para entornos de escritorio. A continuación se muestra el modelo de despliegue de la solución (Ver Figura 14).

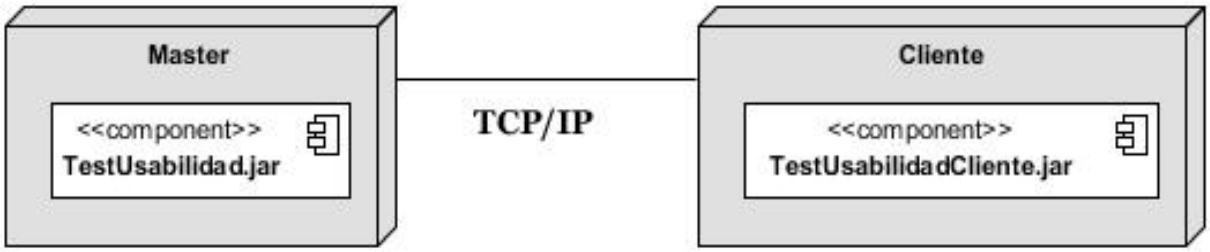


Figura 14: Diagrama de despliegue

Para el despliegue de las aplicaciones es necesario cumplir con una serie de requisitos. Primeramente se debe tener instalado tanto en la máquina máster como en la máquina cliente el sistema operativo Linux, en sus distribuciones Nova 3.0, Nova 4.0, Ubuntu 11.10 y Ubuntu 12.04. Además, se debe contar con la máquina virtual de Java instalada. Luego se define la configuración del entorno de trabajo para cada uno de los ordenadores en los cuales se va a trabajar.

En máquinas que van a ser utilizadas como clientes, se ejecutará la aplicación cliente **TestUsabilidadCliente.jar**. Para este ejecutable funcione correctamente debe configurarse el entorno de trabajo con los siguientes requerimientos:

- Debe estar instalado el programa **ffmpeg**, el cual está liberado bajo licencia GNU y es una completa herramienta para el procesamiento de audio y video, también soporta grabación y codificación en tiempo real. Está desarrollado para el sistema operativo Linux, pero puede ser usado en otros sistemas operativos, incluyendo Windows.
- El ordenador debe contar con 100 MB libres en el disco duro como mínimo.
- Debe contar con una memoria RAM de 256 MB como mínimo.

En las máquinas que van a ser utilizadas como máster, se ejecutará la aplicación máster **TestUsabilidad.jar**. Para este ejecutable funcione correctamente debe configurarse el entorno de trabajo con los siguientes requerimientos:

- Debe estar instalado el reproductor de multimedia VLC Media Player.
- Debe contar con una memoria RAM de 512 MB como mínimo.
- Debe contar con un mínimo de 500 MB libres de disco duro.

Es necesario contar con espacio disponible en el disco duro, debido a que los videos de grabación de uso de los usuarios que realizan el test son almacenados en la máquina máster. Dicha capacidad es variable, ya que dependerá de la cantidad de usuarios a realizar el test y de la cantidad de test a realizar.

Conclusiones parciales

En el presente capítulo se definen las características y principales funcionalidades de la herramienta desarrollada, lo cual posibilitó obtener una herramienta centrada en las necesidades del usuario. Además, se realizó una descripción de la propuesta de acuerdo con los requisitos funcionales a implementar, lo cual permitió la selección de los patrones arquitectónicos y de diseño a utilizar en la propuesta. Se define un estándar de codificación que permitió lograr la uniformidad del código implementado. Se define la situación física de los componentes lógicos desarrollados, modelándose con este objetivo el diagrama de despliegue de la herramienta en cuestión. Se realizó además, el modelado del diagrama de paquetes y de componentes, lo cual permitió lograr un mejor entendimiento de la estructura de la herramienta.

Capítulo 3: Validación de la solución propuesta.

Introducción

Durante el desarrollo del sistema se puede cometer muchos errores por lo que este proceso debe estar acompañado de alguna actividad que garantice el rendimiento y funcionalidad del producto que se le brinda al usuario quedando así lo más libre posible de fallas. En este capítulo se aborda el tema referente a la realización de las pruebas; se realiza un análisis del tipo de prueba a utilizar para validar el software y el diseño de casos de prueba. En este capítulo también se crean los casos de prueba para cada funcionalidad que se han implementado donde se detallan las mismas con sus descripciones generales, condiciones de ejecución, y secciones a probar. Se realiza además el análisis de los resultados obtenidos al finalizar las iteraciones de pruebas.

3.1 Pruebas de software

La etapa de pruebas es una de las fases del ciclo de vida de los proyectos. Se la podría ubicar después del análisis, el diseño y la programación, pero esto depende del proyecto en cuestión y del modelo de proceso elegido. Las pruebas de software no tienen el objeto de prevenir errores sino de detectarlos. Se efectúan sobre el trabajo realizado y se deben encarar con la intención de descubrir la mayor cantidad de errores posibles (SUÁREZ, 2003).Las pruebas a realizar son las que se enuncian a continuación:

Niveles de Prueba:

- Unidades
- Aceptación

3.1.1 Pruebas de unidad

Es la escala más pequeña de las pruebas, está basada en la funcionalidad de los módulos del programa, como funciones, procedimientos y módulos de clases. En ciertos sistemas también se verifican o se prueban los *drivers* y el diseño de la arquitectura (PRESSMAN, 2002). Normalmente, cabe distinguir una fase informal antes de entrar en la fase de pruebas propiamente dicha. La fase informal la lleva a cabo el propio codificador en su despacho, y consiste en ir ejecutando el código para convencerse de que "básicamente, funciona". Esta fase suele consistir en pequeños ejemplos que se intentan ejecutar. Si el módulo falla, se suele utilizar un depurador para observar la evolución dinámica del sistema, localizar el

fallo, y repararlo. Los casos de pruebas que se diseñen para este nivel de pruebas, deben descubrir errores como:

- Comparaciones entre tipos de datos distintos.
- Operadores lógicos o procedencia incorrecta.
- Igualdad esperada cuando los errores de precisión la hacen poco probable.
- Variables o comparaciones incorrectas.
- Fallo de salida cuando se encuentra una iteración divergente.

Se tomaron 15 funciones de cada capa con el objetivo de encontrar algunos de los errores antes mencionados y buscar diversidad en el código a la hora de realizar la prueba. Luego de probadas las unidades, se llegó a la conclusión que funcionaban correctamente, lo cual no quiere decir que todos los métodos del sistema lo hagan de igual forma, para asegurarse de ello es necesario realizar otras pruebas que abarquen más funcionalidades. A continuación se muestra uno de los fragmentos de código probados mediante el método de camino básico (Ver Figura 15).

```
private void jLabel4MouseClicked(java.awt.event.MouseEvent evt) {
    posTabla = tablaTarea.getSelectedRow();                                    1

    if (posTabla > -1) {                                                       2

        if (posTabla > 0) {                                                    3

            confirmarOpcionMoverArriba();                                      4

        } else {

            JOptionPane.showMessageDialog(null, "La  fila  0 no se puede mover hacia arriba",
                    "Error", JOptionPane.ERROR_MESSAGE                         5
                    );

        }                                                                      6
    } else {

        JOptionPane.showMessageDialog(null, "Seleccione la tarea que a eviar arriba ",
                "Error", JOptionPane.ERROR_MESSAGE                             7
                );

    }                                                                          8
}                                                                              9
```

Figura 15: Procedimiento para mover hacia arriba una tarea

Se confeccionó un grafo de flujo con la lógica del código a probar. De esta manera, se podrán determinar todos los caminos por los que el hilo de ejecución pueda llegar a pasar, y por consecuente elaborar los juegos de valores de pruebas para aplicar al módulo, con mayor facilidad y seguridad.

Un grafo de flujo se compone de:

- Nodos (círculos), que representan una o más acciones del módulo.
- Aristas (flechas), que representan el flujo de control entre los distintos nodos.

En este grafo están presentes todos los caminos que pueden guiar la trayectoria del módulo o unidad, de manera que todos sus nodos, excepto el primero y el último, deben tener al menos una entrada y una salida. A continuación se presenta el grafo perteneciente a esta prueba (Ver Figura 16).

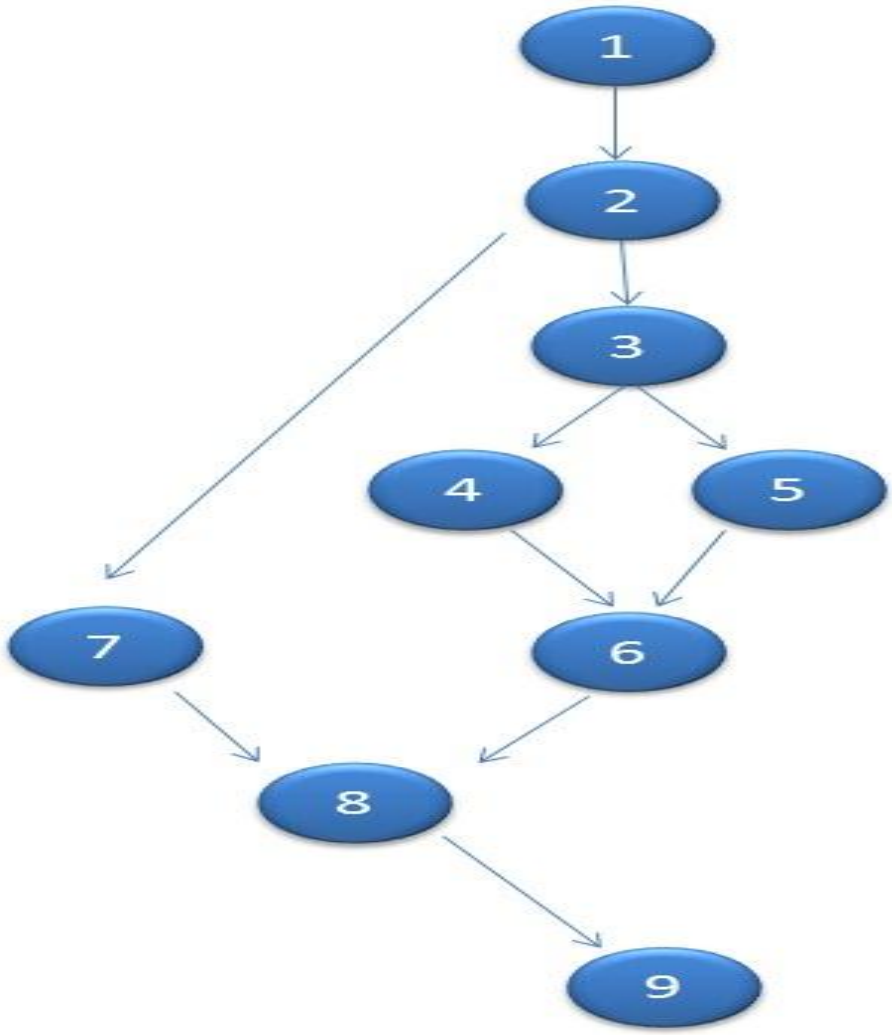


Figura 16: Grafo del camino del módulo a probar.

Este grafo se usa para calcular la complejidad ciclomática del módulo a probar. Para ello son usadas tres vías distintas, cada una de ellas se define por una fórmula, que independientemente de cuál de ellas se use, el resultado siempre debe ser el mismo.

Primera fórmula:

V (G) = (A - N) + 2 (donde A es el número aristas del grafo y N el número de nodos.)

V (G) = (10-9) + 2

V (G) = 3

Segunda fórmula:

V (G) = P + 1 (donde P es la cantidad de nodos desde los cuales se pueden tomar dos caminos.)

V (G) = 2 + 1

V (G) = 3

Tercera fórmula:

V (G) = R (donde R es la cantidad de regiones)

V (G) = 3

Luego de calcular el número de caminos independientes mediante una de las fórmulas, se deberán identificar todos los caminos independientes. Un camino independiente es un camino que introduce un nuevo grupo de acciones o nueva condición.

Caminos independientes: 1,2,7,8,9; 1,2,3,5,6,8,9; 1,2,3,4,6,8,9;

A continuación se preparan los casos de prueba que obliguen a la ejecución de cada camino independiente.

Camino: 1, 2, 7, 8, 9;

Caso de prueba: Mover la tarea hacia arriba.

Entrada: no seleccionar la tarea a mover y presionar clics en el botón arriba.

Salida: muestra un mensaje de error que indica que no se ha seleccionado la tarea.

Camino: 1, 2, 3, 5, 6, 8, 9;

Caso de prueba: Mover la tarea hacia arriba.

Entrada: seleccionar la primera tarea y presionar clics en el botón arriba.

Salida: muestra un mensaje de error que indica que la tarea seleccionada no puede moverse.

Camino: 1, 2, 3, 4, 6, 8, 9;

Caso de prueba: Mover la tarea hacia arriba.

Entrada: seleccionar la tercera tarea y presionar clics en el botón arriba.

Salida: se mueve la tarea hacia arriba y se actualiza en la tabla de tarea.

3.1.2 Pruebas de aceptación

Las pruebas de aceptación se realizan sobre el producto terminado e integrado, están concebidas para que sea un usuario final quien detecte los posibles errores. Se clasifican en dos tipos: pruebas alfa y pruebas beta. Las pruebas alfa se realizan por un cliente en un entorno controlado por el equipo de desarrollo. Para que tengan validez, se debe primero crear un ambiente con las mismas condiciones que se encontrarán en las instalaciones del cliente. Una vez logrado esto, se procede a realizar las pruebas y a documentar los resultados. Cuando el software sea la adaptación de una versión previa, deberán probarse también los procesos de transformación de datos y actualización de archivos de todo tipo. Por otro lado, las pruebas beta se realizan en las instalaciones propias de los clientes. Para que tengan lugar, en primer término se deben distribuir copias del sistema para que cada cliente lo instale en sus oficinas, dependencias y/o sucursales, según sea el caso. En el caso de las pruebas beta, cada usuario realizará sus propias pruebas y documentará los errores que encuentre, así como las sugerencias que crea conveniente realizar, para que el equipo de desarrollo tenga en cuenta al momento de analizar las posibles modificaciones (PRESSMAN, 2002).

3.2 Casos de pruebas

3.2.1 Caso de prueba gestionar tarea del test de usuario.

Escenario	Descripción	Nombre	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Adicionar tarea	La herramienta permite adicionar tareas	V	V	Se adiciona la tarea	1. Clic en adicionar tarea. 2. Se inserta los valores nombre y descripción. 3. Se oprime el botón aceptar
		Tarea #1	Buscar consumo energético		
		V	I	No se adiciona la tarea y se muestra un mensaje “Los	1. Clic en adicionar tarea. 2. Se inserta los valores nombre y
		T1	Vacío		
		I	V		

		Vacío	Buscar el consumo de agua	campos no pueden estar vacíos “.	descripción. 3. Mensaje de error “Los campos no pueden estar vacíos“. 4. Se oprime el botón aceptar
EC 1.2 Editar tarea	La herramienta permite editar tareas	V Tarea #1	V Buscar consumo energético	Se modifica la tarea	1. Clic en editar tarea. 2. Se inserta los nuevos valores nombre o/y descripción. 3. Se oprime el botón aceptar
		V T1	I Vacío	No se modifica la tarea y se muestra un mensaje “Los campos no pueden estar vacíos “.	1. Clic en adicionar tarea. 2. Se inserta los valores nombre y descripción. 3. Mensaje de error “Los campos no pueden estar vacíos“. 4. Se oprime el botón aceptar
		I Vacío	V Buscar el consumo de agua		

Tabla 2. Casos de prueba gestionar tarea del test de usuario

3.2.2 Caso de prueba crear proyecto

Escenario	Descripción	Nombre del proyecto	URL del sitio	Respuesta del sistema	Flujo central
EC 2.1 Crear proyecto	La herramienta permite crear un proyecto	V Intranet	V http://intranet2.uci.cu	Se crea un proyecto	1. Clic en crear proyecto. 2. Se inserta los valores Nombre del proyecto y URL del sitio. 3. Se oprime el botón aceptar
		V	I	No se crear el	1. Clic en crear

		Intranet	Vacío	proyecto y se muestra un mensaje “Los campos no pueden estar vacíos “.	proyecto. 2. Se inserta los valores Nombre del proyecto y URL del sitio. 3. Muestra un mensaje de error "Los campos no pueden estar vacíos" 4. Se oprime el botón aceptar
		I	V		
		Vacío	http://intranet2.uci.cu		

Tabla 3. Caso de prueba crear proyecto

3.2.3 Caso de prueba adicionar PC cliente

Escenario	Descripción	Nombre PC	Dirección IP de la PC	Respuesta del sistema	Flujo central
EC 3.1 Adicionar PC cliente	La herramienta permite adicionar las PC clientes que se conectarán para la ejecución del test de usuario	V	V	Se adiciona PC cliente	1. Clic en adicionar PC clientes 2. Se inserta los valores Nombre PC y Dirección IP de la PC. 3. Se oprime el botón aceptar
		PC1	10.60.69.248		
		V	V		
		Vacío	10.60.69.248		
		V	I	No se adiciona la PC cliente muestra un mensaje de error “La dirección IP no es válida “.	1. Clic en adicionar PC clientes 2. Se inserta los valores Nombre PC y Dirección IP de la PC. 3. Se oprime el botón aceptar 4. Muestra un mensaje de error “La dirección IP no es válida “.
		Usuario	Vacío		
		V	I		
		PC2	0.asd.50.60		
		V	I		
		Vacío	10.60.500.20		

Tabla 4. Caso de prueba adicionar PC cliente

3.3 Resultados obtenidos al finalizar las iteraciones de pruebas.

El siguiente gráfico se representa los resultados obtenidos al finalizar cada iteración (Ver Figura 17).

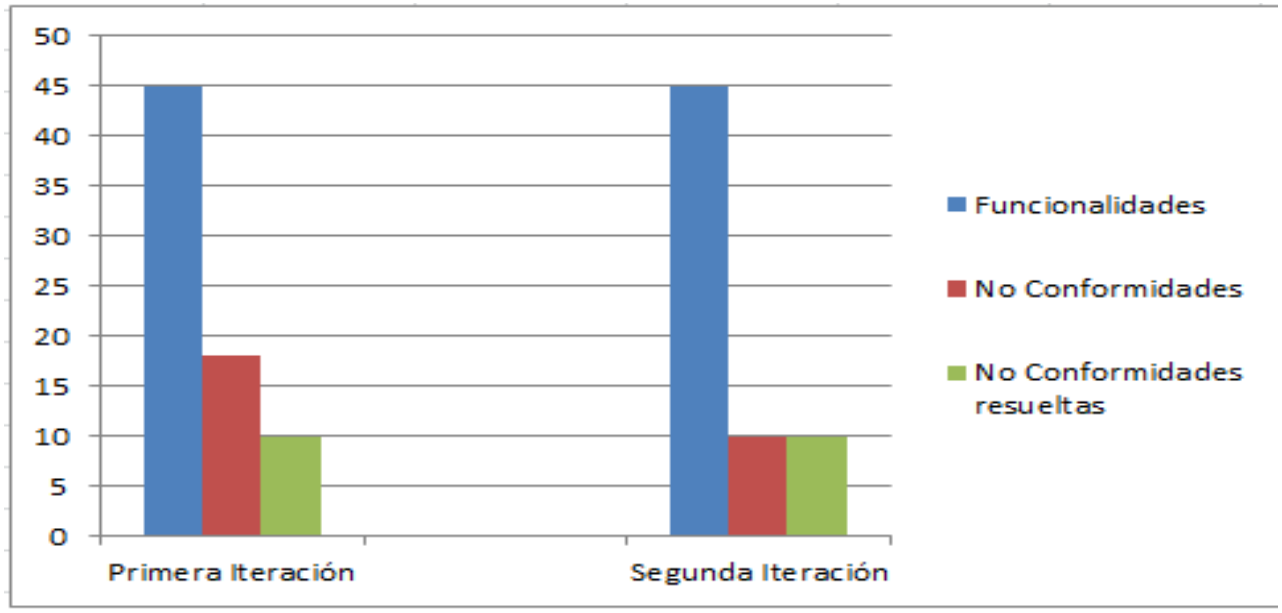


Figura 17: Pruebas del sistema

En la primera iteración de pruebas al sistema de 45 funcionalidades se encontraron 18 no conformidades las cuales se clasifican en 12 significativas y 6 no significativas, de ellas se le dio solución a 12. En la segunda iteración se detectaron 10 no conformidades, de ellas 8 significativas y 2 no significativas las cuales se pudo resolver en su totalidad.

Conclusiones parciales

En el presente capítulo fue establecida una estrategia de validación que permitió certificar el correcto funcionamiento de la herramienta desarrollada, demostrando que sus funcionalidades satisfacen las necesidades del cliente y concuerdan con los requisitos definidos en la etapa inicial del proyecto. Como resultado final de la presente investigación se obtuvo la versión 1.0 de la herramienta para el análisis de la usabilidad mediante la aplicación de la técnica de test de usuario apoyada por el método de evaluación grabación de uso.

Conclusiones generales

Con la realización del presente trabajo de diploma se le dio cumplimiento al objetivo general planteado, destacándose de manera general las siguientes conclusiones:

- Se realizó la fundamentación teórico-metodológica de la investigación, permitiendo conceptualizar el proceso de evaluación de la usabilidad mediante la técnica de test de usuario.
- Se realizó un estudio de las soluciones actuales que utilizan la técnica test de usuario y otros métodos de evaluación de la usabilidad permitiendo identificar aspectos de interés para el desarrollo de la propuesta.
- Se realizó el diseño arquitectónico de la propuesta, permitiendo definir la estructura de la herramienta.
- Se realizó la definición de un estándar de codificación permitiendo la organización y uniformidad en el código.
- Se comprobó la capacidad y potencialidad de la herramienta a través de la estrategia de validación trazada en la presente investigación, garantizando que su código fuente presentase la calidad requerida.

Recomendaciones

- Incorporar a la herramienta los métodos de evaluación de la usabilidad que no fueron desarrollados en la presente investigación, como son mapa de clics, hablando en voz alta y encuesta.
- Incorporar a la herramienta el módulo de análisis heurístico propuesto por la analista.

Bibliografía referenciada

ALONSO, D. E. Y. A., P. L. ObRemUs: Una Herramienta para la Observación Remota de Usuarios. 2008, nº Disponible en: http://www.aipo.es/info_art.php?id=73.

ALTOFT, P. *SEO and Eye Tracking for Informational & Transactional Queries.* Disponible en: http://www.branded3.com/blogs/seo-and-eye-tracking/.

ATTENTIONWIZARD. *Consultora de Experiencia de Usuario* Disponible en: www.attentionwizard.com/.

BEVAN, N. What is Usability? 1991, nº Disponible en: www.nigelbevan.com/papers/whatis92.pdf.

CARROLL., R. Usability Engineering. 2002, nº

CHALKMARK. Consultora de Experiencia de Usuario. 2012, nº Disponible en: http://www.optimalworkshop.com/chalkmark.htm.

CHILE., G. D. Guía para el desarrollo de páginas web del Gobierno de Chile. "Capítulo 5: De la Usabilidad a la Utilidad. 2008, nº Disponible en: http://www.guiaweb.cl/guia-v2/capitulos/05/.

CLICKDENSITY. *Consultora de Experiencia de Usuario* Disponible en: www.clickdensity.com.

CRAZYEGG. *Consultora de Experiencia de Usuario* Disponible en: www.crazyegg.com/.

DIOSDADO, S. *Eye-tracking de las redes sociales.* Disponible en: http://www.samueldiosdado.com/01/eye-tracking-de-las-redes-sociales/.

ESTÁNDAR_DE_CÓDIGO. Estándar de Código. 2010, nº Disponible en: https://repositorio.cenia.prod.uci.cu/svn/documentacion/Dpto_UD/ABAD/EP_ABAD/4.general/CENIA_G
AI_OT_Estándar _de _Código.pdf

FLOWER, M. *Patterns of Enterprise Application Architecture*. En 2003.

GARCÍA, G. L. R., E. I. Propuesta de un manual para los planificadores de proyectos productivos en la UCI. 2009, nº

GRANOLLERS, T. L., JESÚS. PERDRIX, F. *Incorporación de Usuarios en la Evaluación de la Usabi-lidad por el Recorrido Cognitivo.* 2004.

GRIHO. Modelo de Proceso de la Ingeniería de la usabilidad y de la accesibilidad. MPIu+a 2010 nº Disponible en: http://www.grihohcitools.udl.cat/mpiua/index.html.

HASSAN MONTERO , Y. M. F., FRANCISCO J. IAZZA, GHZALA. Diseño Web Centrado en el Usuario: Usabilidad y Arquitectura de la Información. 2004, nº Disponible en: http://www.hipertext.net/web/pag206.htm. ISSN 1695-5498.

HASSAN, M. Y. S. H., V. Eye-Tracking en Interacción Persona-Ordenador." No Solo Usabilidad(6). 2007, nº

HASSAN, Y. M., F. J. No solo usabilidad. Qué es la Accesibilidad Web. 2003, nº Disponible en: http://www.nosolousabilidad.com/articulos/accesibilidad.htm.

IEEE. *Software Engineering Standards*
IEEE Computer Society, 1994,

JACOBSON, I., BOCH, G., & RUMBAUGH, J. *El Proceso Unificado de Desarrollo de Software.* 2000.

KRUCHTEN, P. *Architectural Blueprints--The 4+1 View Model of Software Architecture.* 1995.

KUCHANA, P. software Arquitecture Design Patterns in Java. 2004, nº

LARMAN, C. UML y Patrones. 1999, nº

LETELIER, P. P., Mª CARMEN.
Métodologías ágiles para el desarrollo de software:eXtreme Programming (XP). nº

LOOP11. *Consultora de Experiencia de Usuario* Disponible en: www.loop11.com/.

LORENZO ESCOBAR, A. M. *Propuesta de herramienta para análisis de usabilidad en el paquete Abad.* Universidad de las Ciencias Informáticas, 2012.

MERCER, D. Fundamentos de Programación en XML. 2001, nº Disponible en: http://bibliodoc.uci.cu/pdf/reg01311.pdf.

NIELSEN, J. Usability Engineering. 1993, nº

NIELSEN , J. Usability Return on Investment. 2002, nº Disponible en: www.nngroup.com/reports/roi/.

PALACIO, J. Flexibilidad con Scrum. 2008, nº

PREECE, J. *Interaction Design: Beyond Human-Computer Interaction.* 2 edition ed. 2007. ISBN 978-0470018668.

PRESSMAN, R. S. *Ingienería de Softare.Un enfoque práctico.* Quinta edición. ed. 2002.

PROJECT, U. N. E. U. 2003, Disponible en: http://www.usabilitynet.org.

RODRÍGUEZ, D. Y. B., EDUARD. . *usabilidad* Disponible en: http://www.trucosoptimizacion.com/index.php/2011/02/02/test-remoto-usuarios-3-herramientas-cambiaran-usabilidad/.

SHRUM, S. Y. K., M. *CMMI : GUIA PARA LA INTEGRACION DE PROCESOS Y LA MEJORA D E PRODUCTOS.* 2da ed. ADDISON-WESLEY, 2009. ISBN 9788478290963.

SILVA, E. Criterios de Usabilidad en la Web. 2009, nº Disponible en: http://www.face.uc.edu.ve/depardeportes-uc/bibliografia/.

SILVERBACK. Disponible en: http://silverbackapp.com/.

SINHA, R. Prácticas Ágiles-Desarrollo de software con un enfoque ágil. 2010, nº

STANDARDIZATION., I. O. F. Guidance on usability ISO 9241-11. 1998, nº

---. ISO 9241-210. 2010, nº

SUÁREZ, P. Documentación y Pruebas. 2003, nº

USERZOOM. *Consultora de Experiencia de Usuario* Disponible en: http://www.userzoom.com/.

VEGA LEBRÚN, C. S. G., ARTURO. Mejores Práctica para el establecimiento y aseguramiento de la Calidad de Software. 2000, nº

WOODSON, W. E. Human Factors Design Handbook. 1981, nº

Bibliografía consultada

AVELEIRA, R. Y. Paquete de herramienta de software para apoyar el diseño de experiencia de usuario. 2012.

AVELEIRA, R. Y. y BARRERA, D. S. Laboratorio para diseño de experiencia de usuario. Revista Cubana de Ciencias Informáticas, 2012, vol. 5, nº 3, ISSN 2227-1899.

BIAS, R. G. y MAYHEW, D. J. Cost-Justifying Usability: An Update for the Internet Age. 2 ed. Estados Unidos: Morgan Kaufmann, 2005. ISBN 978 -0120958115.

CERTUCHE, J. A.; ZAPATA, R. D. O., et al. Técnicas de usabilidad y accesibilidad orientadas a procesos de desarrollo de software. En I Congreso Internacional Computación y Matemática. Costa Rica. Agosto 2008.

CONSTANTINE, L. L. y LOCKWOOD, L. A. D. Software for use: a practical guide to the models and methods of usage-centered design. SI GCHI Bulletin, January 2000, vol. 32, nº 1, Disponible en: http://bulletin.sigchi.org/2000/january/news -events -publications/Constantine.pdf.

COSTABILE, M. F. Usability in the software life cycle: Handbook of software engineering and knowledge engineering, 2001, vol. 1, nº p. 179-192.

ESCALONA, C., MARÍA JOSÉ y GONZÁLEZ, R., JOSÉ MA RIANO Diseño Centrado en el Usuario. España: Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla, 2009, 36 p. Disponible en: http://www.lsi.us.es/docencia/master_its/.

FERRÉ, X.; JURISTO, N., et al. Framework for integrating usability practices into the software process. Lecture Notes in Computer Science, 2005, vol. 3547, nº p. 202- 215. Disponible en: http://www.springerlink.com/content/y6debb8l25p9lvqh/.

FERRERAS, B. y JACQUELÍN, H. Aplicación de la usabilidad al proceso de desarrollo de páginas web. Informática, 2008.

FREIRE, Á. V. y KAFURE, I. et al. Propuesta para la detección de aspectos subjetivos en los métodos de valuación de usabilidad. 2012, nº Disponible en: http://dilnxsrv.king.ac.uk/lacnem2012/PastProceedings/lacnem2011/papers/pre1.pdf.

FUENTES, R. A. Metodología de evaluación de usabilidad para aplicaciones web transaccionales. 2008.

GARCÍA-MIRELES, A.; GARCÍA, F., et al. Influencia de la calidad del proceso en la usabilidad del producto: una revisión sistemática. 2011, nº Disponible en: http://lbd.udc.es/jornadas2011/actas/JISBD/JISBD/S4/Regulares/jisbd2011_submission_17.pdf.

GONZÁLEZ, V., D. ¿Qué es la Experiencia del Usuario?: Nethodical.com. Disponible en, 2004.

KAFURE, I. Usabilidad y diseño emocional en la gestión de la información. En 2009.

LIM, K, Y. y LONG, J. B. The MUSE method for usability engineering. Cambridge University Press, 2009. vol 8, ISBN 0521479991.

MONTES, D. O. S. D. B., A. Arquitectura de información y usabilidad: nociones básicas para los profesionales de la información. Acimed, 2004, vol. 12, nº 6, p.1- 1. ISSN 1024-9435.

STEWART, T. Usability or user experience what's the difference. System concepts [On-line]. Disponible en: http://www.system-concepts.com/articles/usability%20&%20hci/usability%20or%20user%20experience. 2008, vol. 20, nº 11.

Glosario de términos

Drivers: es un programa que controla un dispositivo, ya sea una impresora, un teclado, entre otros.

HTML: es un lenguaje creado con el fin de visualizar e interconectar el contenido de documentos electrónicos, por lo que consideró un conjunto pequeño de etiquetas que marcaran párrafos, títulos, hipervínculos y entre otros elementos.

ISO: Organización Internacional de Normalización, es el organismo encargado de promover el desarrollo de normas internacionales de fabricación (tanto de productos como de servicios), comercio y comunicación para todas las ramas industriales a excepción de la eléctrica y la electrónica. Su función principal es la de buscar la estandarización de normas de productos y seguridad para las empresas u organizaciones (públicas o privadas) a nivel internacional.

ISO/IEC 9126: es un estándar internacional para la evaluación de la calidad del software.

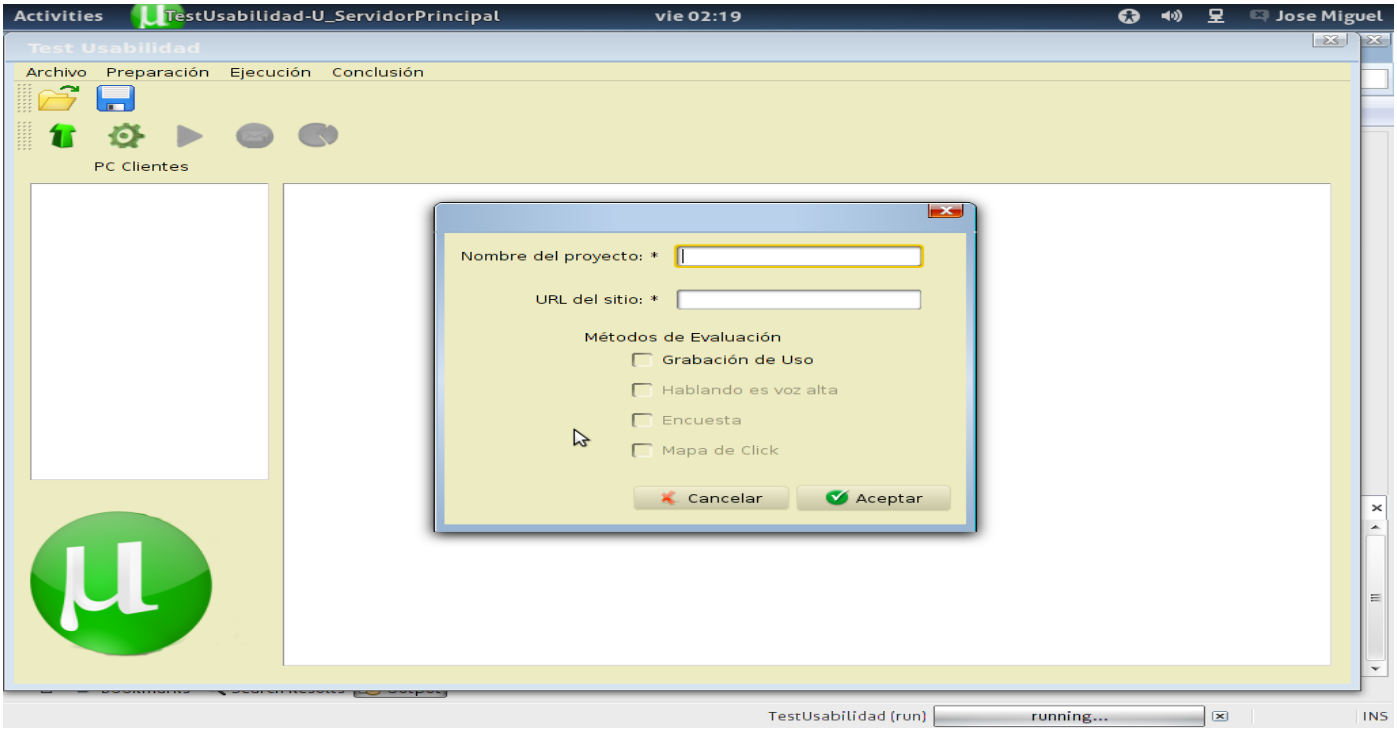
Java: lenguaje de programación orientado a objetos.

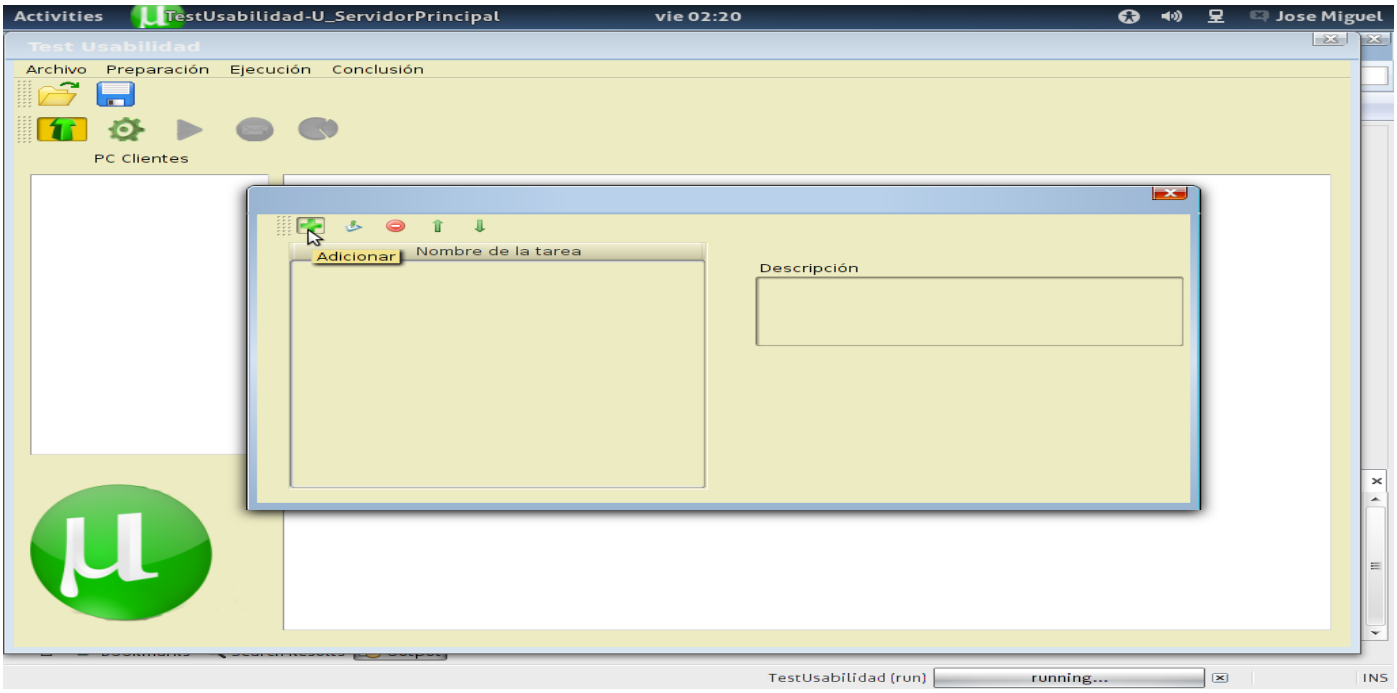
Usabilidad: facilidad de uso, ya sea de una página web, una aplicación informática o cualquier otro sistema que interactúe con un usuario.

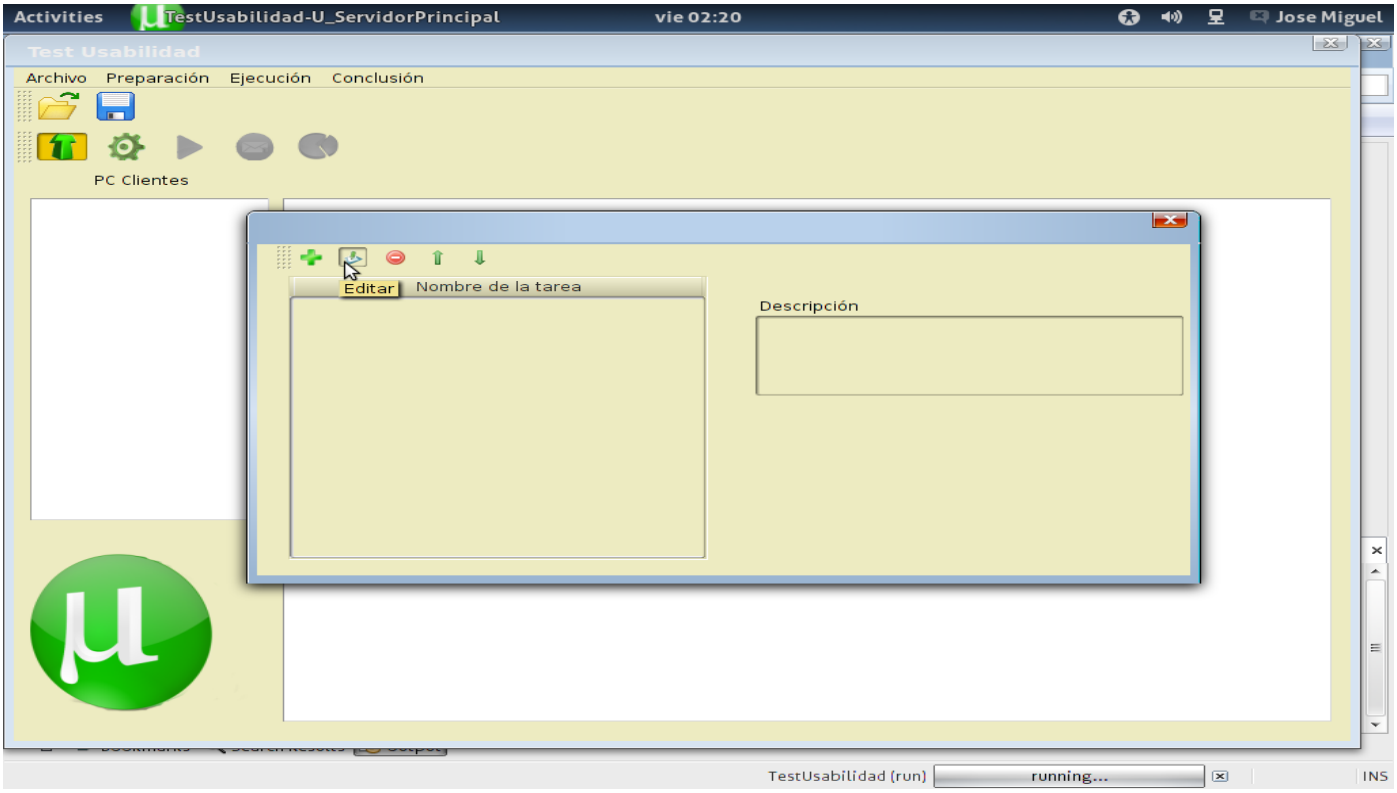
XML: Lenguaje de Marcas Extensible, por su traducción al español. Es un metalenguaje -lenguaje que describe los datos y cómo estos se estructuran- mediante el cual, los desarrolladores pueden crear sus propios elementos para alcanzar sus propias necesidades de información.

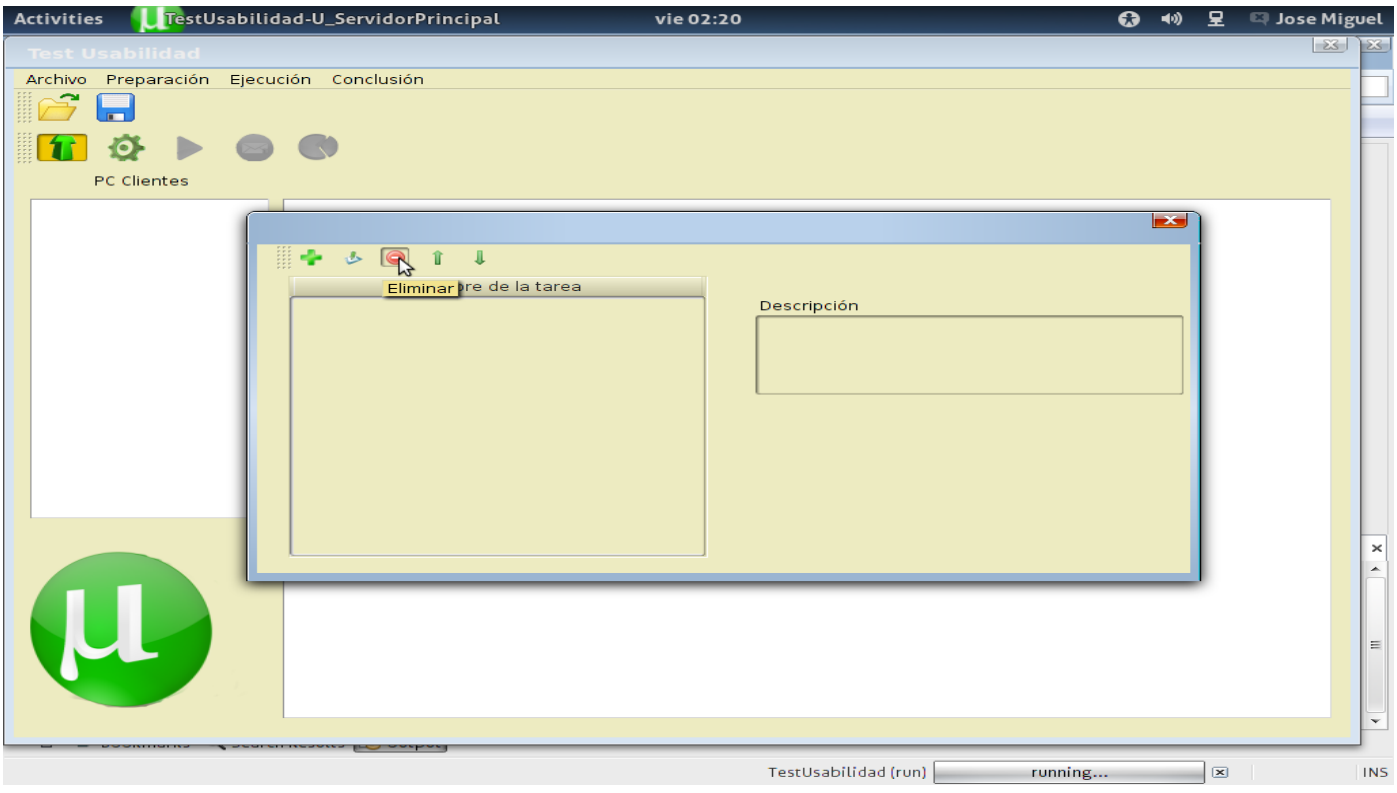
Anexos

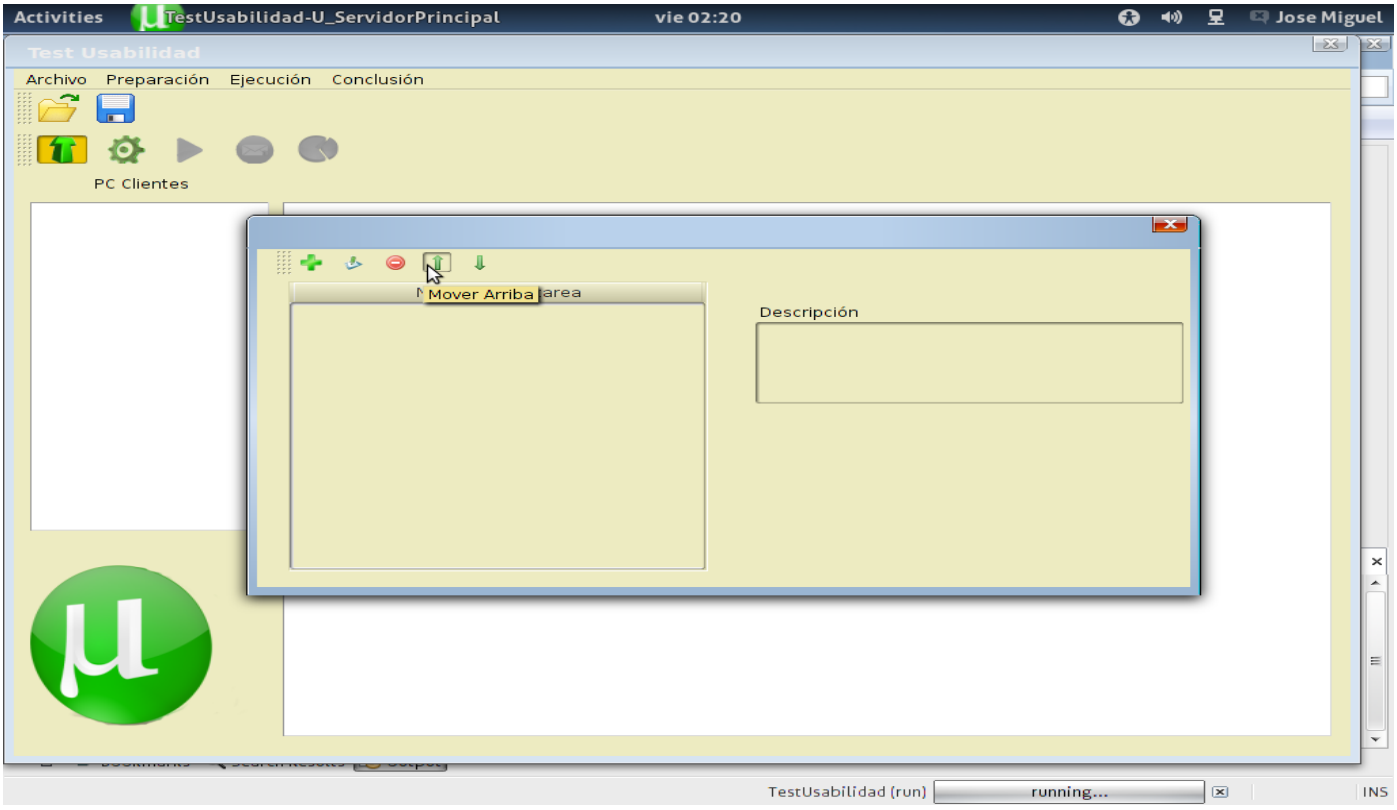
Anexo 1 Interfaces del paquete Preparación

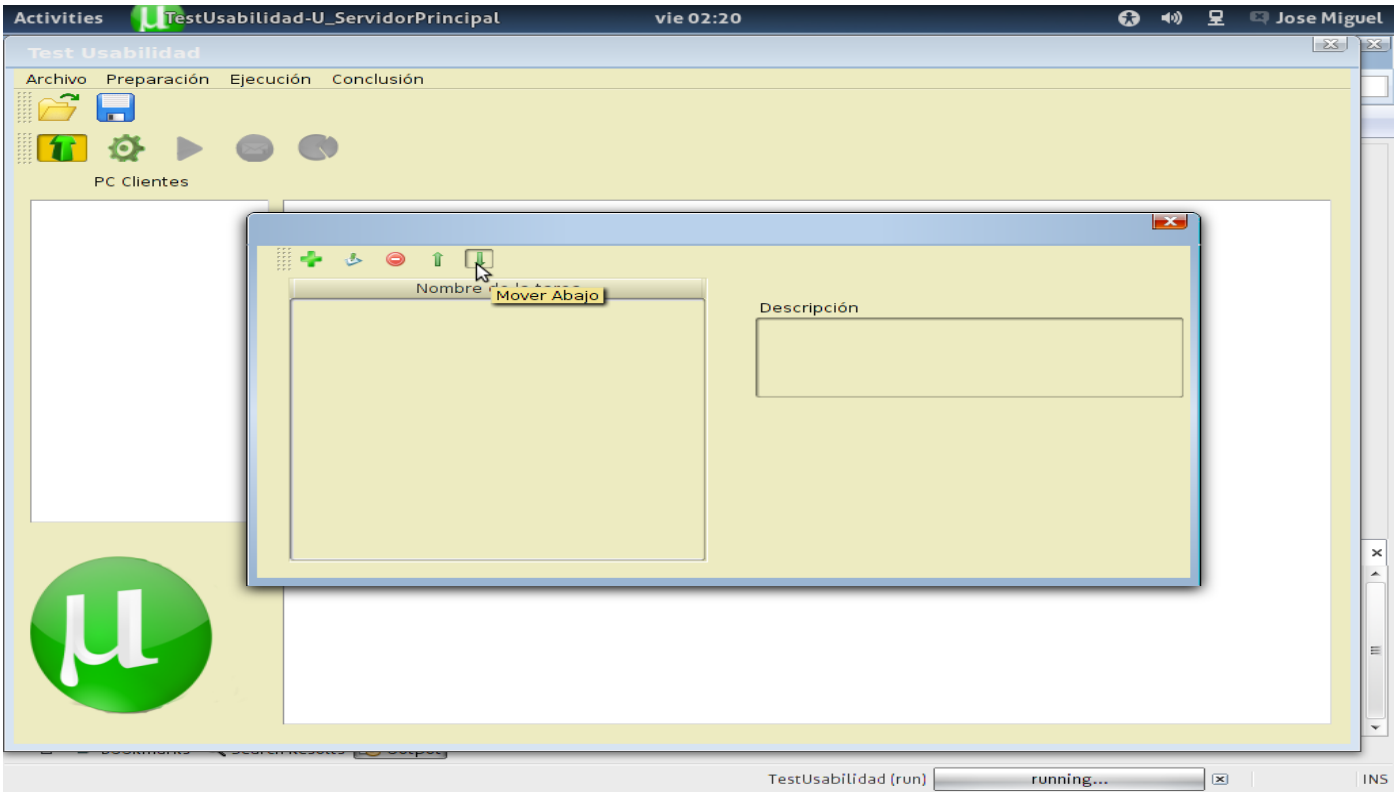


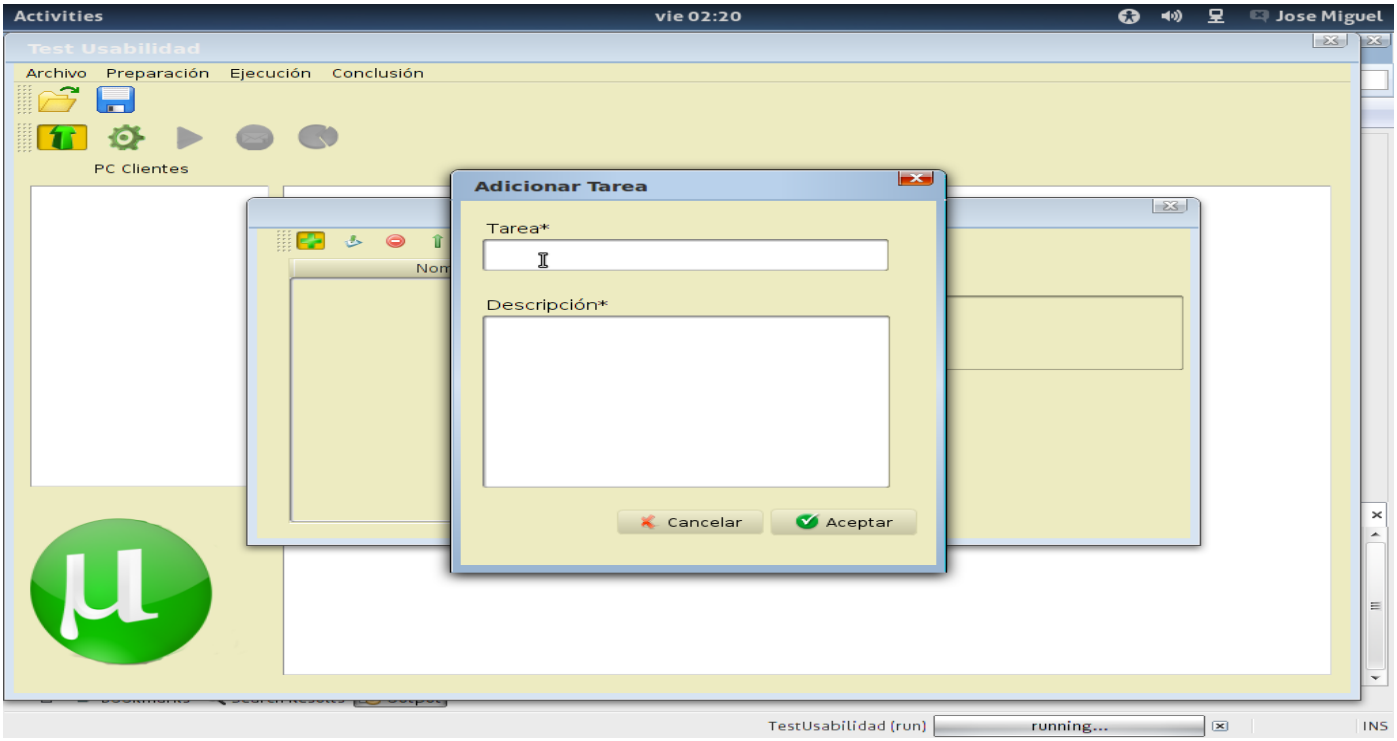



Activities
TestUsabilidad-U_ServidorPrincipal
vie 02:20
Jose Miguel
Test Usabilidad
Archivo
Preparación
Ejecución
Conclusión
PC Clientes
Editar
Nombre de la tarea
Descripción
TestUsabilidad (run)
running...
INS

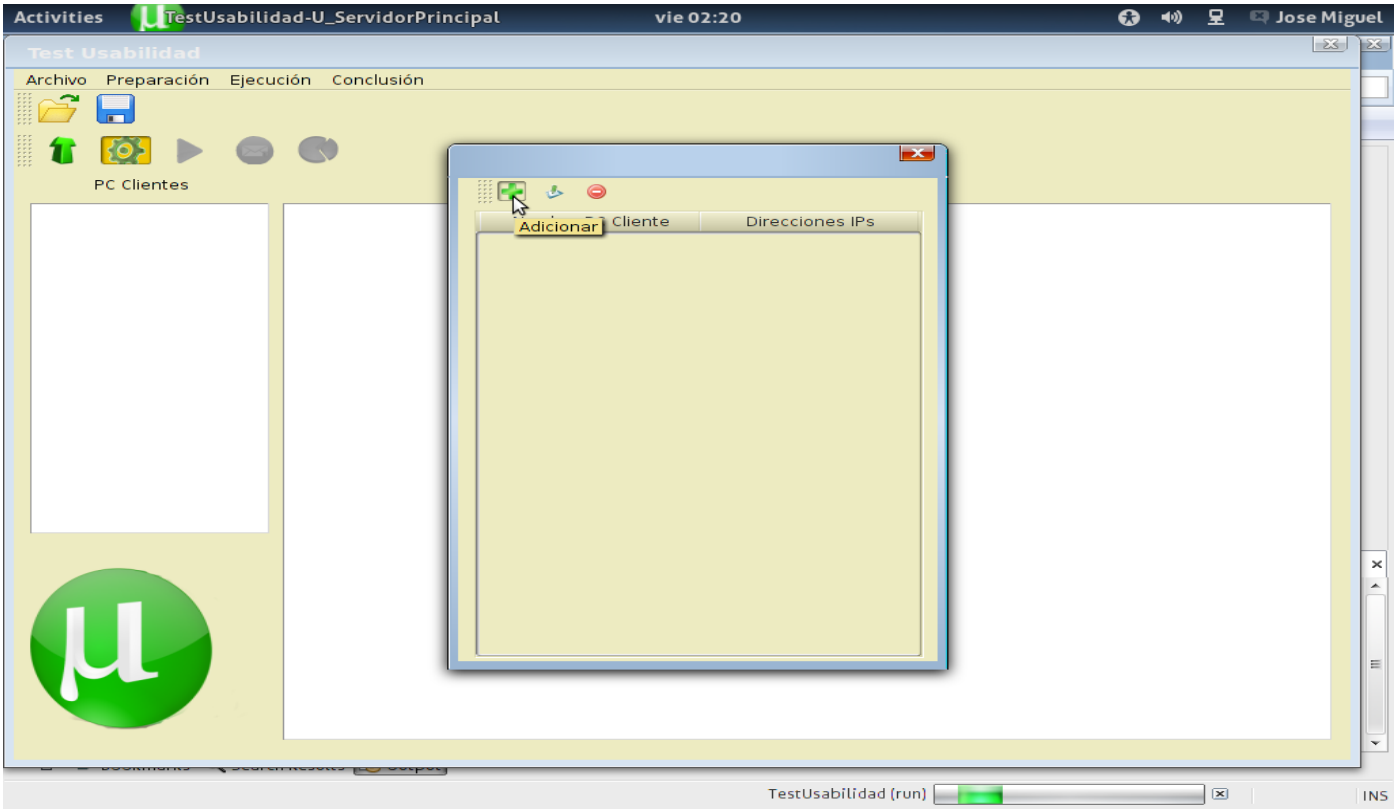

Activities
TestUsabilidad-U_ServidorPrincipal
vie 02:20
Jose Miguel
Test Usabilidad
Archivo
Preparación
Ejecución
Conclusión
PC Clientes
Eliminar
Descripción
TestUsabilidad (run)
running...
INS

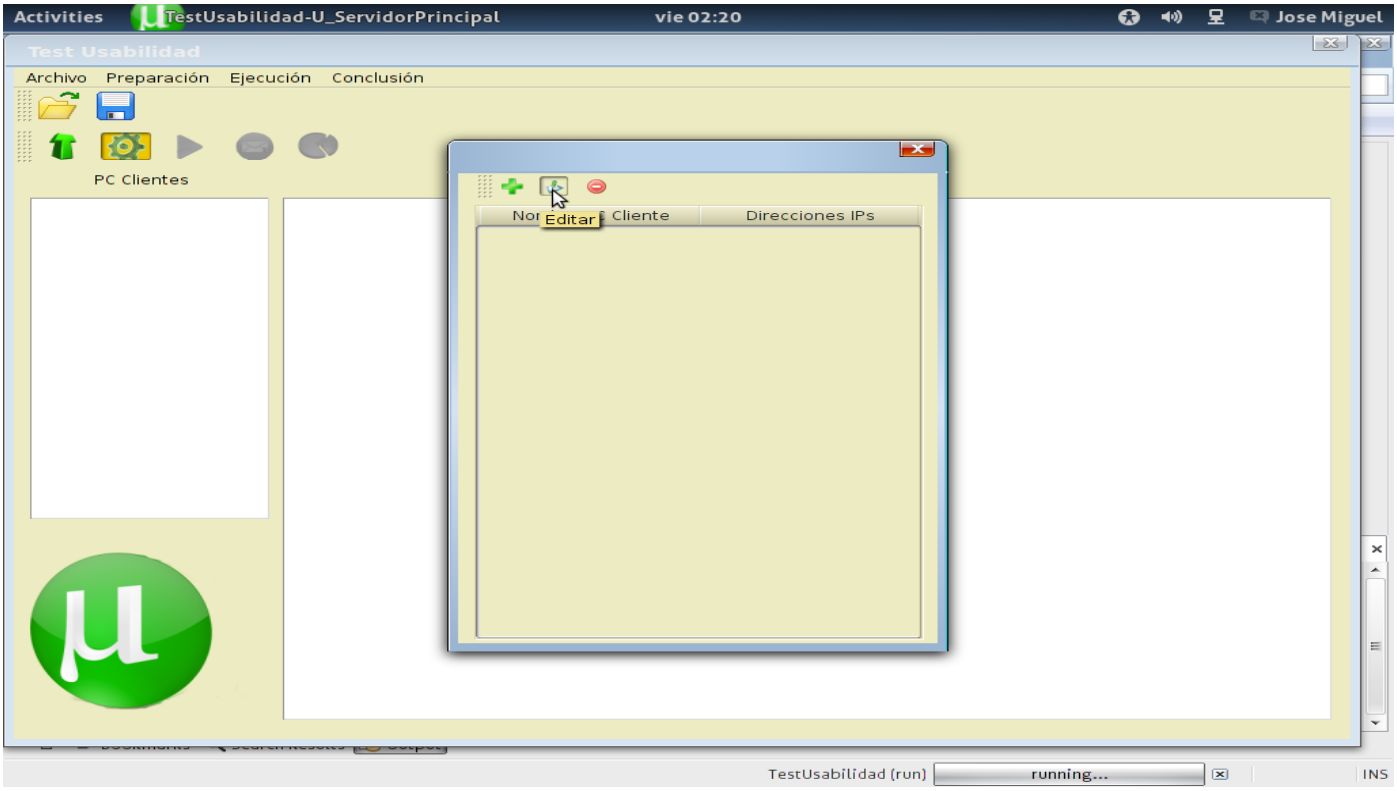

Activities
TestUsabilidad-U_ServidorPrincipal
vie 02:20
Jose Miguel
Test Usabilidad
Archivo
Preparación
Ejecución
Conclusión
PC Clientes
Mover Arriba
Descripción
TestUsabilidad (run)
running...
INS

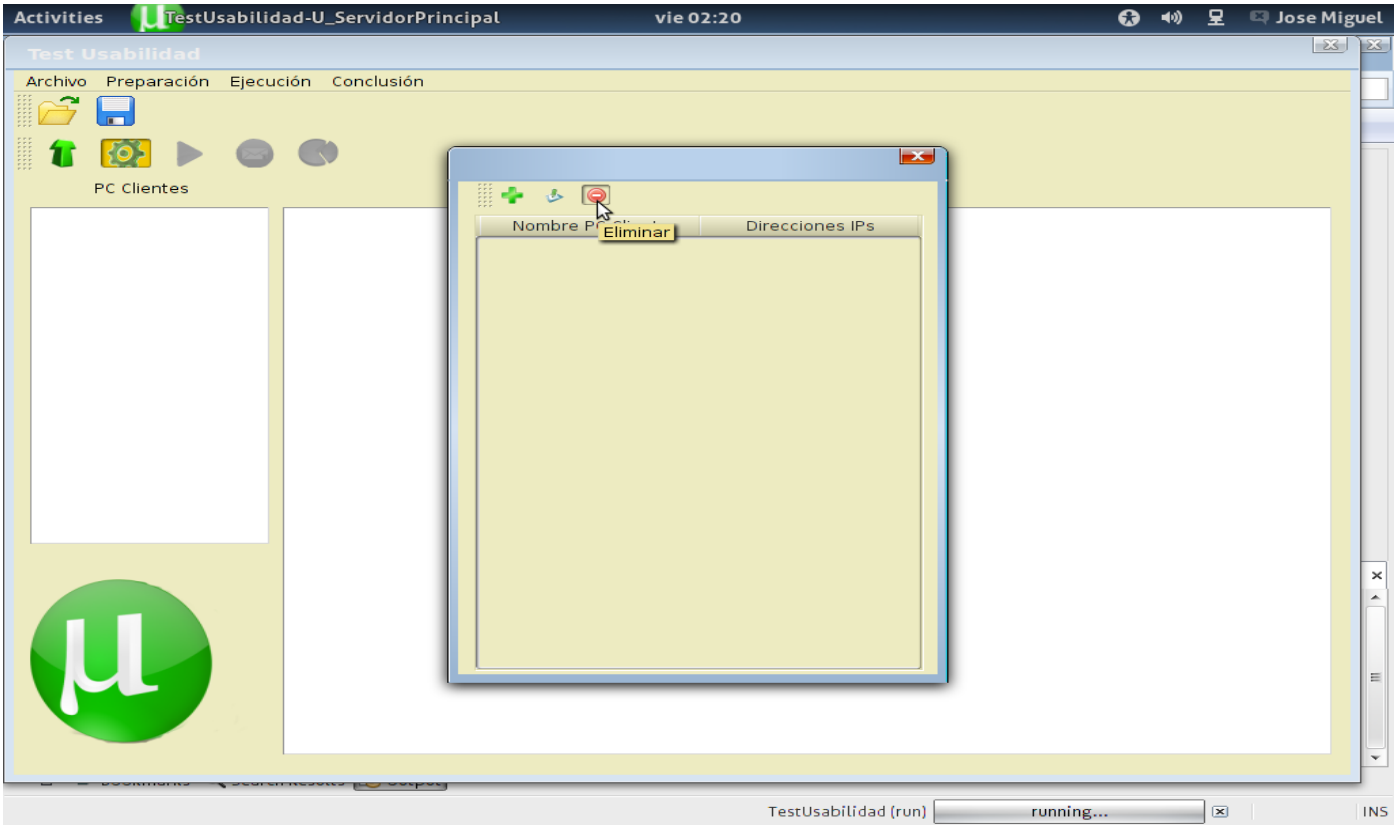

Activities
TestUsabilidad-U_ServidorPrincipal
vie 02:20
Jose Miguel
Test Usabilidad
Archivo
Preparación
Ejecución
Conclusión
PC Clientes
Mover Abajo
Descripción
TestUsabilidad (run)
running...
INS

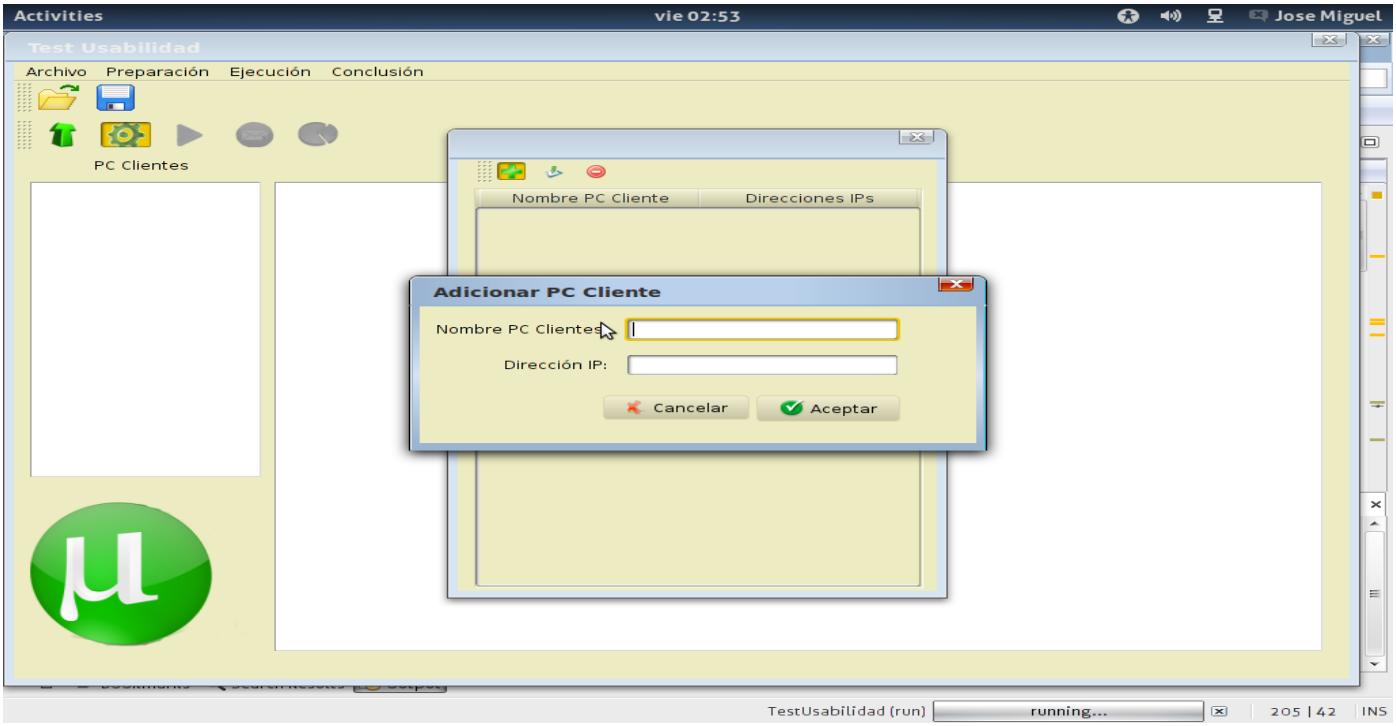


Anexo 2 Interfaces del paquete Ejecución

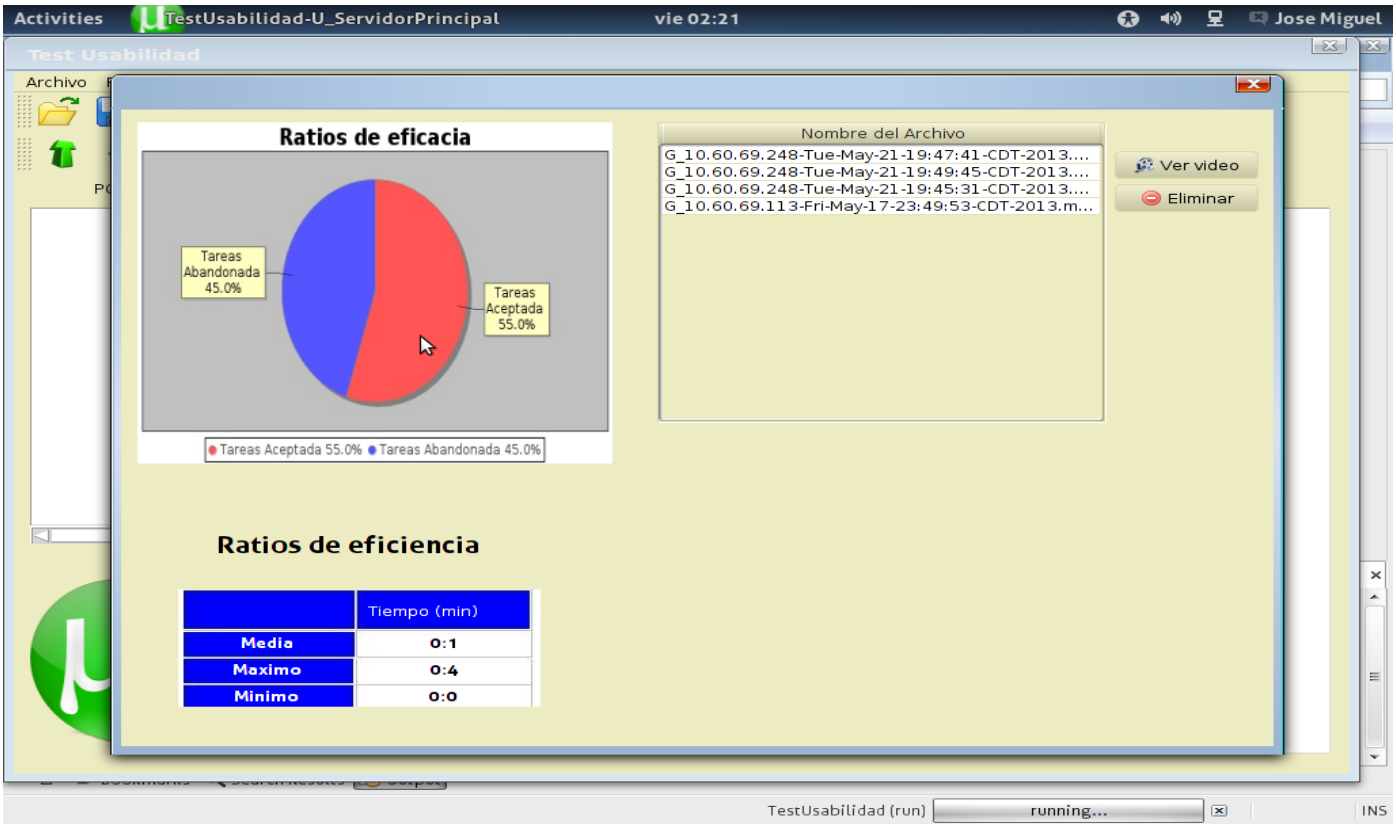



Activities
TestUsabilidad-U_ServidorPrincipal
vie 02:20
Jose Miguel
Test Usabilidad
Archivo
Preparación
Ejecución
Conclusión
PC Clientes
Editar
Direcciones IPs
TestUsabilidad (run)
running...
INS

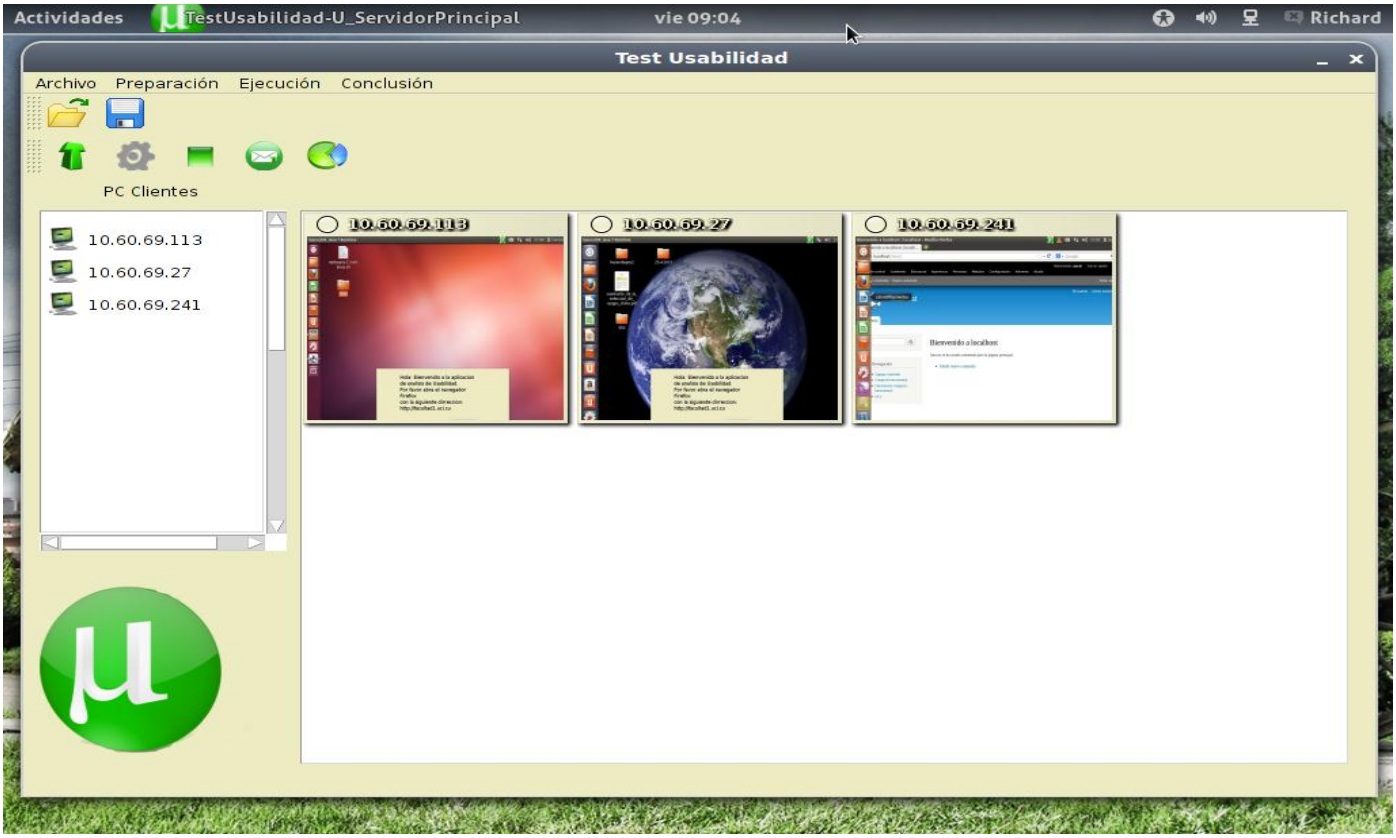

Activities
TestUsabilidad-U_ServidorPrincipal
vie 02:20
Jose Miguel
Test Usabilidad
Archivo
Preparación
Ejecución
Conclusión
PC Clientes
Eliminar
Direcciones IPs
TestUsabilidad (run)
running...
INS



Anexo 3 Interfaces del paquete Conclusión



Anexo 4 Interfaces del paquete Paneles




¡Bienvenido al sitio de la Biblioteca de la Universidad de las Ciencias Informáticas! | Dirección de Informa
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
¡Bienvenido al sitio de la Bibliotec...
biblioteca.uci.cu
Google
Ingresar a su cuenta
UCI
Dirección de Información
Inicio
Quiénes Somos
Servicios
Catálogo
Repositorio Institucional
Bases de Datos
Tutoriales
Publicaciones Seriadas
Boletín Informativo CESIM
Boletín Teduscopio FORTES
Revista Ciencias de la Información
Revista Cubana de Ciencias Informáticas
Revista La jiribilla
Revista Temas
Serie Científica
Vigitec
Otros Recursos de Información
arXiv.org
¡Bienvenido al sitio de la Biblioteca de la Universidad de las Ciencias Informáticas!
La Dirección de Información de la Universidad de las Ciencias Informáticas les "proporci
productos de información de alto valor agregado que apoyen las funciones principales de
(docencia, investigación, extensión universitaria y producción) mediante la selección ad
fuentes de información y la utilización de las tecnologías de la información y las comunicaci
Noticias
Homenaje a José Martí
Nuevas Adquisiciones
Nuevas Adquisiciones
Estimados usuarios, ponemos en su conocim
donaciones recibidas durante la pasada edici
Internacional del Libro de La Habana. Entre l
recibidos se encuentran:
Tarea # 1/1
Buscar un libro por su autor
Aceptar
Abandonar