Guía de actuación en el desarrollo de interfaces de usuario según la metodología centrada en el usuario INTERGRAM

Begoña Losada
Depto de Lenguajes y Sistemas Informáticos
Facultad de Informática
Universidad del País Vasco
20018 San Sebastián
b.losada@ehu.es

David López
Facultad de Informática
Universidad del País Vasco
20018 San Sebastián
dlopez024@ikasle.ehu.es

Javier Martínez
Facultad de Informática
Universidad del País Vasco
20018 San Sebastián
jmartinez059@ikasle.ehu.es

Resumen

El diseño de interfaces realizado con criterios de calidad exige un proceso de desarrollo con modelos centrados en el usuario en todas sus etapas, desde la descripción del perfil de usuario hasta la evaluación de una aplicación interactiva en uso. La aplicación de métodos de ingeniería se facilita con el uso de herramientas que abstraigan su utilización y guíen el proceso. El presente artículo describe los pasos y conexiones de la metodología INTERGRAM y muestra la herramienta Intergram basada en dicha metodología. Nos enfocaremos principalmente en los módulos conceptuales de esta utilidad. Esto es, por una parte, Diagram, que recoge la captura de requerimientos, permite la descripción del diálogo H-M y la transcribe en un documento UIML (User Interface Markup Language); y, por otra parte, el módulo evaluador Evalgram que permite, tomando como soporte la descripción en UIML, producir prototipos primitivos para realizar evaluaciones tempranas con los usuarios potenciales.
Palabras clave. Ingeniería de la interfaz, metodología, herramientas de desarrollo de interfaces de usuario

1. Introducción

El desarrollo de software es considerado como una ingeniería en el sentido de que debe realizarse mediante la aplicación estructurada de técnicas científicas. De la misma forma, el desarrollo de interfaces de usuario se convierte en una ingeniería al utilizar para la realización de un producto técnicas específicas de ingeniería. Es precisamente en la utilización de estas técnicas en lo que se basa la calidad de un producto software interactivo [7] [14] [15] [17] [18] [24] +[25]. Como indica Olsina en su tesis [20], la utilización sistemática de métodos, modelos y técnicas de Ingeniería de Software para la evaluación, control y mejora de la calidad de los productos Web debería ser un requerimiento obligatorio en todo proyecto de mediana o gran escala.

Dentro de las metodologías para el desarrollo de interfaces, se impone desde hace tiempo aquéllas en las que el usuario se implica en todo el ciclo de vida del diseño [30] y que se agrupan bajo el nombre de metodologías centradas en el usuario. Dentro de éstas, la metodología INTERGRAM se caracteriza por un proceso de evaluación continuo, la inclusión de un orden riguroso de actuación y por las traslaciones a modelos descriptivos de la información elaborada en las distintas fases.

La conveniencia de una herramienta que facilite y guíe la actuación de INTERGRAM motiva el diseño de Intergram, cuya descripción conceptual es otro de los objetivos de este artículo.

Para cubrir estos aspectos, el resto del artículo se estructura de la siguiente manera: en el segundo apartado se muestran los ciclos de vida clásicos en ingeniería de software comparándolos con los modelos centrados en el usuario en el desarrollo de interfaces. Después, se describe el esquema propuesto en la metodología INTERGRAM. A continuación, se describe la herramienta Intergram, sus módulos y conexiones. Los siguientes apartados ahondan en la explicación de la subherramienta Diagram que utiliza modelos

formales mediante los que el diseñador representa el comportamiento de los usuarios y del sistema en el nuevo escenario, y del módulo evaluador Evalgram que utiliza técnicas de prototipado temprano para una evaluación automatizada con los usuarios. Terminaremos con unas conclusiones de nuestras aportaciones en el desarrollo de interfaces de usuario.

2. El ciclo de vida en el desarrollo de interfaces

Un modelo de ciclo de vida muestra las fases significativas o las actividades de una proyecto software desde su concepción hasta que finaliza. En este apartado comenzaremos por describir los modelos clásicos en el desarrollo de software, seguiremos con los modelos en el desarrollo de interfaces y finalmente, describiremos el ciclo de vida propuesto en INTERGRAM.

2.1. El ciclo de vida en el desarrollo de software

Desde la ingeniería de software, se han hecho numerosos intentos por encontrar el modelo óptimo en el desarrollo de software.

El ciclo de vida clásico de la ingeniería de software es el modelo en cascada en el que las actividades se enlazan secuencialmente y en donde cada fase finaliza completamente antes de que comience la siguiente [27] (Figura 1)

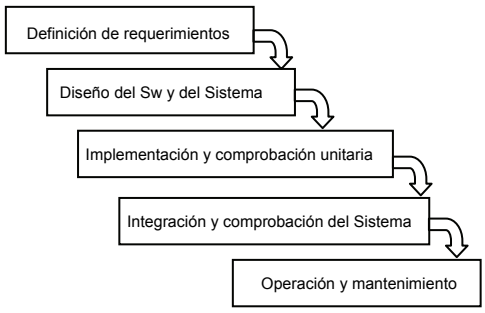

Figura 1. Modelo en cascada

Durante la especificación de los requerimientos, el diseñador y el usuario tratan de hacer una descripción de qué proporcionará el nuevo sistema. Las siguientes actividades se centran en cómo proporciona el sistema lo que se espera de él mediante el diseño estructural y un diseño detallado de los componentes. La codificación y comprobación traslada lo descrito en la etapa anterior a algún lenguaje ejecutable. Después, los componentes son testados para verificar su buen funcionamiento según algunos criterios determinados en etapas previas. Finalmente, los distintos componentes deben ser integrados según dicta lo descrito durante el diseño estructural.

Stone [30] prueba que en la práctica, las fases de desarrollo se solapan y se proporcionan información unas a otras. Durante el diseño, son identificados problemas en los requerimientos; durante la codificación, se detectan problemas de diseño…El proceso de software no es una modelo lineal simple sino que requiere una secuencia de iteraciones en las actividades de desarrollo.

Esta visión simple es ampliada por el mismo autor [28] añadiendo iteraciones hacia delante y hacia atrás entre los distintos pasos.

Un modelo evolucionado a partir del modelo en cascada es V-Shaped [23] en el que destacan planificaciones para evaluaciones tempranas.

El modelo en espiral [2] tiene cuatro fases: planificación, análisis de riesgos, ingeniería y evaluación. Un proyecto software pasa muchas veces por estas fases visualizado en una espiral que va ampliándose en número de iteraciones y coste.

El modelo DSDM (Dynamic Systems Development Method) [29] define una fase de preparación, estudio de viabilidad y estudio de negocio secuencialmente para la adquisición de normas para el resto del desarrollo. Esto es: Iteración del modelo funcional, Iteración del diseño y construcción y finalmente implementación. Termina con una fase de conclusión en la que se deja la solución operativa. Se caracteriza por su naturaleza iterativa e incremental. El mantenimiento es una parte más del ciclo de vida.

2.2. El diseño centrado en el usuario

En ISO 13407 p.7 [11] encontramos una guía de principios en el diseño centrado en el usuario, para conseguir sistemas usables:

- Participación activa de los usuarios
- Distribución apropiada de funciones entre el usuario y el sistema
- La iteración de las soluciones de diseño
- Equipos de diseño multidisciplinares

Según establece ISO 13407 p.10 [12], las cuatro actividades de diseño centradas en el usuario esenciales serían:

- Entender y especificar el contexto de uso
- Especificar los requerimientos de organización y de usuario
- Producir soluciones de diseño (prototipos)
- Evaluar diseños con usuarios y comparar con los requerimientos

La diferencia principal entre el ciclo de vida clásico y el diseño centrado en el usuario es que en éste el diseño se centra en el usuario, su trabajo y su entorno, y en cómo la tecnología puede ser desarrollada y diseñada para soportarlo [24]. Además, se caracteriza por un proceso de diseño rápido e iterativo con evaluaciones continuas. Es, por lo tanto, necesario el uso de representaciones centradas en el usuario.
Como muestra de la evolución en los ciclos de vida clásicos en el desarrollo de interfaces centrados en el usuario, Greenberg [8] (Figura 2) propone un proceso iterativo de diseño de interfaces y evaluación

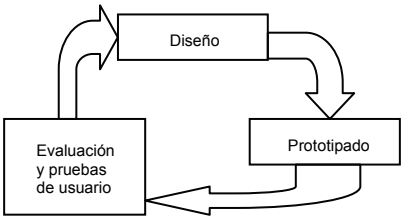

Figura 2. Proceso de diseño y desarrollo de interfaces de usuario de Greenberg

El ciclo de vida en estrella [10] (Figura 3) centra en la evaluación el punto central de todas las etapas. Todos los aspectos del sistema son sujetos a evaluación constante por usuarios y expertos. El proceso es completamente iterativo, el desarrollo del sistema puede empezar en cualquier fase y a ésta puede seguirle cualquier otra, explicado por la evolución constante de los requerimientos, el diseño y el producto hasta una escritura definitiva.

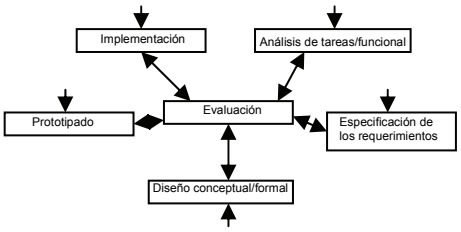

Figura 3. Ciclo de vida en estrella

2.3. La metodología INTERGRAM

En la línea de los diseños centrados en el usuario, nuestra propuesta (Figura 4) incide, como el modelo de Hix mostrado en la figura 3, en la integración del proceso de evaluación en todas las etapas del ciclo de vida y no sólo al final como ocurre en el ciclo clásico en cascada. Incluye igualmente entre sus premisas la necesidad de prototipado rápido. El esquema propuesto se caracteriza también por una fuerte iteración en el proceso. Compartimos con los modelos tradicionales las etapas básicas: análisis (análisis de tareas/análisis funcional), especificación de los requerimientos, diseño (diseño conceptual/diseño formal) e implementación. Añadimos etapas de prototipado y evaluación como en el modelo de Hix.

Siguiendo con el propósito genérico de los ciclos de vida vistos, nuestra metodología es válida para cualquier software interactivo. Es adecuada para su utilización en diseño web pero no restringe su uso a este campo.

INTERGRAM difiere en la inclusión de un orden riguroso de actuación que permite mantener un flujo de información entre los módulos, alimentados y actualizados constantemente por esos flujos en pasos iterativos.

La inclusión de una codificación intermedia permite mantener la independencia de los modelos

utilizados, de las herramientas utilizadas en el diseño y del dispositivo al que va dirigido el producto. Actúa como punto central de almacenamiento de la información, punto de llegada y de partida de todas las fases. Permite recuperar un producto ya realizado y efectuar modificaciones sobre los modelos de concepción primitivos.

El proceso está inmerso en una disciplina de evaluación progresiva que condiciona las iteraciones. Distinguimos dos tipos de evaluaciones: Evaluaciones con usuarios realizadas con prototipos mediante técnicas de prototipado temprano y evaluaciones heurísticas y métricas realizadas por el sistema de acuerdo a criterios establecidos en el modelo del sistema.

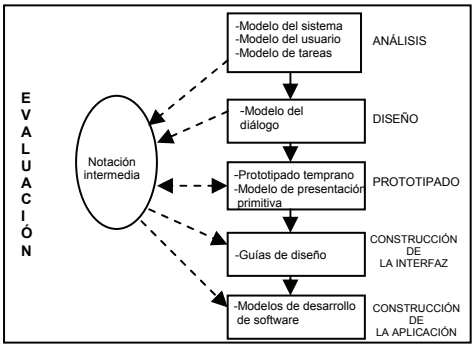

Figura 4. Proceso en INTERGRAM

3. La herramienta Intergram

La herramienta Intergram guía al diseñador en los pasos de actuación descritos en la metodología INTERGRAM y facilita la utilización de modelos de ingeniería abstrayendo las notaciones complejas. Existen numerosas herramientas software de generación de interfaces de usuario basadas en modelos [16]. En Intergram se han tenido en cuenta las siguientes premisas que la caracterizan:

- Sigue una metodología de diseño centrada en el usuario según se ha explicado en el apartado anterior. Esto ayudará a producir diseños bien evaluados y en definitiva a realizar trabajos usables.
- Facilita el trabajo del diseñador guiando el proceso.
- Utiliza técnicas y modelos formales de ingeniería de la interfaz para la realización de los procesos descritos en INTERGRAM. Esto permite conseguir diseños de calidad, con una base firme.
- Existe una documentación en un formato específico de XML actualizada continuamente durante el desarrollo que permitirá retomar un diseño, modificarlo y reutilizarlo.

El esquema de Intergram aparece descrito en la figura 5. Esta utilidad dispone de un fondo de almacenamiento organizado que actúa como representación intermedia, y tres módulos: Diagram, Evalgram y Logram, que permiten organizar la labor del diseñador en tres roles: diseñador, evaluador y administrador. Describimos estos módulos a continuación:

- Diagram: Es la la herramienta que utiliza modelos formales con los que el diseñador representa la forma en que los usuarios van a interactuar en el nuevo escenario. Su descripción detallada aparece en el apartado 4.
- Evalgram: Produce el storyboard del sitio interactivo a partir de los modelos anteriores, así como prototipos tempranos para su evaluación por los usuarios. Su descripción detallada aparece en el apartado 5.
- Logram: Actúa cuando la aplicación ya está en uso. Recoge la información almacenada del diseño y los logs de uso, realizando una evaluación de la interfaz en uso en comparación con el diseño previsto. Durante la puesta en marcha del producto, se recogen datos reales. Se utilizan para ello aplicaciones de captación de la interacción. Mediante aplicaciones espías en sistemas monopuesto o bien analizando los logs del servidor para un sitio Web se pueden obtener datos cuantitativos del éxito de un sitio. Indican qué páginas son las más demandadas, qué versiones de navegador (en su caso) se utilizan más... pero no se pueden detectar puntos de fracaso según los caminos de actuación prevista. Las aplicaciones de análisis utilizadas por nosotros obtienen las interacciones detalladas de cada usuario y las comparan con la navegación descrita en la representación intermedia recogida en el fichero UIML. El análisis sintáctico de los

mismos destaca los fracasos en los objetivos de cada usuario y las formas de actuación de los distintos tipos de usuarios ante los objetivos previstos. Todo ello redundará en una mejora de la usabilidad en escenarios reales.

Las representaciones utilizadas por los diseñadores describen de una manera gráfica la actividad lógica de los usuarios. Esta sintaxis suele ser más fácil de interpretar y ejecutar que una textual. INTERGRAM utiliza por tanto de cara a los diseñadores modelos gráficos pero conversiones textuales para lograr objetivos generales de independencia y reutilización, así como objetivos concretos para la recogida de la información añadida en etapas de diseño de la interacción y presentación, y para la recuperación de dicha información en etapas de análisis y evaluación posterior.

Partiendo de los modelos de diseño, y de manera automática, deducido de la realización de los pasos anteriores y actualizado con las sucesivas evaluaciones tempranas del prototipo, se obtiene una descripción textual de la aplicación interactiva con técnicas formales. Ello permite de una parte, describir la navegación de manera precisa, detallando todas las interacciones posibles a partir de un punto y por tanto los recorridos previstos por el diseñador; y de otra parte, incluir las características de los elementos de la interacción presentes en el diseño que se van adaptando según se evalúan con los usuarios. Nuestra propuesta incluye una representación textual intermedia del diseño interactivo que puede ser especificado en una traslación directa desde los modelos descritos en el punto 4.1, permaneciendo implícito mediante una descripción de los modelos empleados con XML, en concreto UIML [1] descrito brevemente en el punto 6.

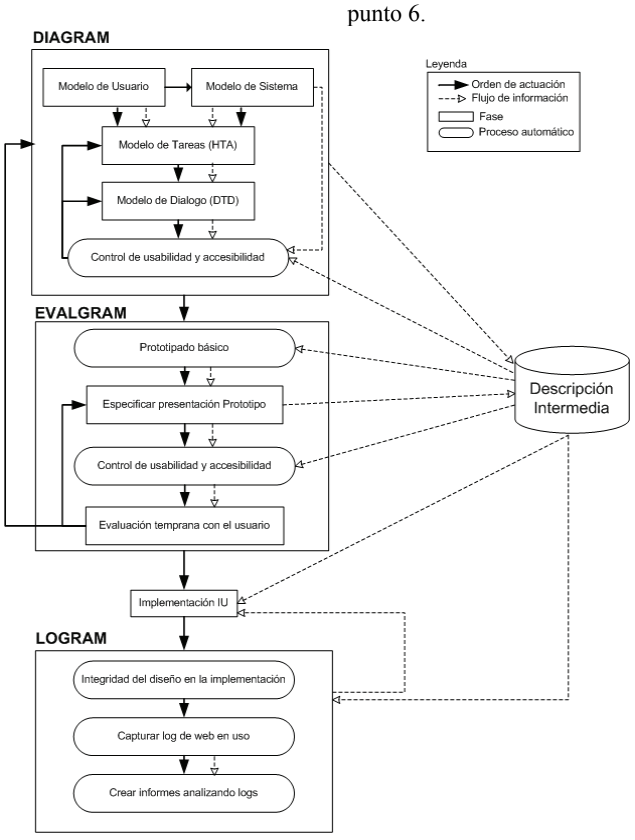

Figura 5. Esquema de Intergram

4. Diagram

La herramienta Diagram facilita al diseñador la concepción del nuevo escenario a través de métodos formales de alto nivel. Su objetivo es dotar a las siguientes fases de la metodología Intergram de una especificación completa de la interfaz de usuario a nivel abstracto. Para ello, tres son las tareas básicas que el diseñador puede realizar:

1. Representar el HTA
   El modelo HTA (Hierarchical Task Analysis) empleado nos permite describir las acciones lógicas que realizará el usuario en el nuevo escenario desmenuzando las tareas en subtareas que completan un objetivo de usuario.
2. Representar el DTD
   El modelo DTD (Dialog Transition Diagram) describe los objetos interactivos y sus relaciones, dando lugar a los estados del diálogo y sus condiciones de cambio.
3. Comprobaciones heurísticas sobre el DTD
   Sobre el esquema de navegación definido en el diálogo, ya es posible realizar comprobaciones heurísticas, sobre todo, en lo referente al control y libertad del usuario.

4.1. Modelando las tareas

La correspondencia entre el nuevo escenario y el mundo real, es decir, seguir el orden lógico del usuario para lograr un objetivo, es un aspecto muy importante para el éxito del diseño [19]. Por ello, Diagram permite al diseñador representar las conclusiones de su trabajo previo de captura de las tareas analizando el comportamiento del usuario. Una vez representado, deberá ser consultado durante el diseño del diálogo ya que es la referencia sobre la que debe basarse.

Existen numerosos trabajos que documentan distintos modelos y variantes para la realización de este proceso: entre ellos, el modelo UAN (User Action Notation) [9] en el que se muestran las acciones de usuario que causan cambios de estado en la interfaz en una sintaxis textual, el modelo GOMS (Goals, Operators, Methods & Selection Rules) [3], en el que se describen las acciones necesarias para lograr un objetivo (goal) y se agrupan en métodos que pueden tener asociada una regla de selección.

La notación CTT (ConcurTaskTrees) desarrollada por Paternò [22], surgió con el objetivo de ser una notación sencilla, con capacidad para modelar tareas cooperativas y multiusuario. En este sentido, coincidimos con ellos y con Nielsen [17] [18] en la conveniencia de usar métodos más intuitivos en unas herramientas que utilizarán desarrolladores y diseñadores no expertos en investigación HCI, por lo que la notación HTA [14] en combinación con notación JSD (Jackson Structured Diagram) [13] para representar tareas repetitivas o disyuntivas, nos parece la elección adecuada en nuestras aplicaciones para la realización del análisis de tareas.

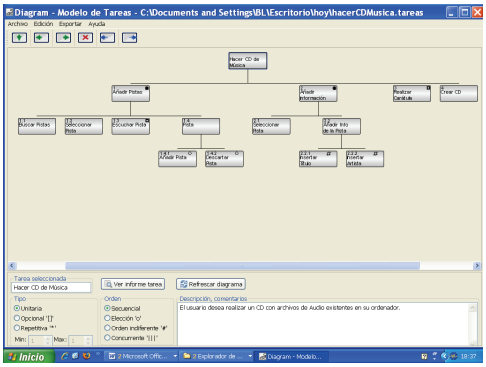

Figura 6. Ejemplo de un proceso de diseño con DIAGRAM

4.2. Modelando el diálogo

La interacción persona-ordenador en sistemas web o multimedia está basado en turnos. Un estado, se corresponde a una pantalla y lo que en ella se ofrece al usuario determina su interacción, su voluntad de llevar la conversación de un estado a otro en función de sus objetivos.

Para modelar el diálogo y siguiendo con el mismo objetivo de uso no experto, optamos por el uso de diagramas de transición de diálogo (DTD) basados en los diagramas de transición de estados (STD) [21]. El diseñador, al igual que

para la notación HTA, puede introducir estos diagramas de una manera intuitiva y visual.

4.3. Heurísticos sobre el DTD

Uno de los campos de la sociología trata sobre el análisis del diálogo. A pesar de estar centrado en lenguaje natural, algunos de sus aspectos pueden extrapolarse al diálogo persona-ordenador. Tras el análisis, el diseñador posee, de forma automática, datos sobre el número de pantallas, el número máximo y medio de objetos interactivos por pantalla, distancias máximas y medias entre pantallas etc., y aparte, puede recibir avisos sobre el posible restricción del control y la libertad del usuario [19]. Es decir, la existencia de pantallas sin retorno, o, teniendo retorno posible, la voluntad del usuario de volver a su estado anterior implica un camino considerablemente largo.

El objetivo de aplicar heurísticos sobre el DTD consiste en facilitar información sobre el diálogo al diseñador, para que pueda tomar decisiones de modificación sobre el mismo.

5. Evalgram, prototipado y evaluación

La herramienta *Evalgram* es un pilar básico dentro de la metodología propuesta en Intergram. Su misión principal es llevar a cabo la evaluación temprana de prototipos de interfaces Web con usuarios potenciales, pudiendo diferenciar dos grandes etapas dentro del ciclo iterativo:

- Fase de Prototipado, en la cual se determina la apariencia o diseño físico de la interfaz en desarrollo.
- Fase de Evaluación, donde a través del feedback proporcionado por un potencial usuario, se evalúa la validez del prototipo diseñado.

El propósito perseguido es el desarrollo rápido de interfaces, agilizando su prototipado y con ello, conseguir ajustarse lo más fielmente posible al modelo mental de los usuarios destinatarios de la interfaz en cuestión, logrando finalmente una navegación satisfactoria en términos de usabilidad.

5.1. Prototipado, especificando el modelo de presentación

En esta fase, se deberán detallar los distintos aspectos relativos al modelo de presentación del prototipo, y mediante él, probar ideas de diseño con usuarios y poder reunir sus impresiones [24].

Cuando se utiliza tempranamente en el proceso de desarrollo, un prototipo alienta la participación e implicación del usuario, y permite a los desarrolladores observar el comportamiento de los usuarios y su reacción ante el prototipo [10].

Inicialmente, la descripción intermedia en formato UIML generada por *Diagram* sólo contiene aspectos relativos a la navegación. Por ello, es necesario determinar el modelo de presentación, *Evalgram* transformará dicho documento de manera automática para generar el prototipo básico relativo a esta descripción y obtener de igual manera el *storyboard navegacional*, sin necesidad de que el diseñador intervenga en este proceso.

En cuanto, al *storyboard navegacional (o de uso)*, podemos destacar, que esta técnica combina las virtudes del *storyboard*, permitiendo representar los diferentes estados de los interfaces, propuesto por [4] y la capacidad de representación de los *mapas de navegación entre contextos* [5], en los cuales se establece la arquitectura general de la Interfaz de Usuario modelando las relaciones entre contextos de interacción. Ello permite visualizar a grandes rasgos el flujo de trabajo de los usuarios en el sistema prototipado.

Por otro lado, está el prototipado de papel o Paper Prototyping [26], especialmente apropiado para etapas tempranas de diseño y su virtud reside en su alto poder de rápida modificación y cuenta con la ventaja de que son mucho más baratos de elaborar que los prototipos software, a no ser que se disponga de una herramienta de prototipado rápido como la proporcionada por Evalgram, ofreciendo un prototipo primitivo sobre el cual determinar el modelo de presentación más específicamente, gracias a la descripción intermedia proporcionada por Diagram.

De esta manera, el diseñador únicamente centrará su actividad en el correcto

funcionamiento de los distintos aspectos estéticos relativos al prototipo, ajustando convenientemente aquellos parámetros relativos a la apariencia del mismo (posicionamiento, dimensiones, colores empleados y metáforas presentadas por los elementos contenidos en el interfaz). Consiguiendo rápida y eficazmente, en las sucesivas iteraciones de revisión, lograr la máxima aproximación al modelo mental del usuario.

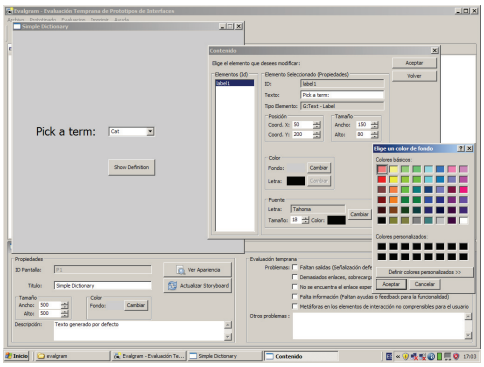

Figura 7. Ejemplo de un proceso de evaluación con EVALGRAM

5.2. Evaluación temprana, pruebas con el usuario

Una vez culminada la fase de prototipado, es necesario llevar a cabo el proceso de evaluación "temprana" centrado fundamentalmente en 3 aspectos según Dix [6].

1. Comprobar la extensión de la funcionalidad del sistema.
2. Comprobar el efecto de la interfaz en el usuario.
3. Identificar cualquier problema específico con el sistema.

El diseño deberá adaptarse a los requisitos y necesidades de los usuarios, por ello es necesario comprobar el efecto que el prototipo genera en el sujeto, pudiendo así localizar aquellos aspectos que generen desconcierto y por tanto susceptibles de ser mejorados.

Evalgram promueve un diseño centrado en el usuario, por ello, se proporciona la capacidad de realizar la navegación del prototipo por parte de un potencial usuario, mediante un simulador, posibilitando la evaluación de la interfaz diseñada, y obteniendo las conclusiones pertinentes referentes al comportamiento del usuario ante el sistema prototipado y poder subsanar aquellos aspectos mejorables en sucesivas iteraciones.

Dentro de las diversas técnicas relativas a los tests de usabilidad, Evalgram se centra fundamentalmente en la aplicación de *la técnica "Thinking aloud" o Pensar en Voz Alta* [19], donde la idea primordial es conseguir la opinión del usuario mientras usa el sistema para evitar posteriores racionalizaciones. Las valoraciones proporcionadas por el usuario deberán ser convenientemente recogidas por el evaluador en la herramienta, determinando así aquellos puntos problemáticos para posteriormente dentro del ciclo iterativo, reajustar de nuevo el diseño y volver a comprobar de nuevo el comportamiento del usuario ante el prototipo mejorado.

El análisis relativo al comportamiento del prototipo recogido por el evaluador, será integrado por Evalgram dentro de la descripción intermedia, del mismo modo, que el modelo de presentación del prototipo es incorporado a dicho documento.

De esta manera, el documento descriptivo quedará convenientemente especificado, sirviendo de referencia al diseñador en iteraciones posteriores, ya que podrá realimentar la descripción con aquellas mejoras sobre el prototipo que considere oportunas.

6. El uso interno de UIML para la descripción de la interfaz

Intergram es una aplicación de alto nivel que se sustenta en un lenguaje que permite describir interfaces complejas. Realizado un estudio en el que barajamos múltiples opciones, desde gramáticas formales (idea inicial) hasta lengua jes declarativos basados en XML, fueron estos últimos la solución adoptada coincidiendo con la tendencia actual en este campo.

Concretamente, UIML (User Interface Markup Language) [1] fue la opción escogida por estar formalmente especificado, tener una estructura sencilla, permitir una descripción abstracta y poseer un vocabulario genérico con elementos interactivos ya definidos.

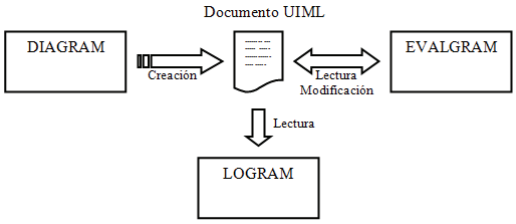

Figura 8. Uso del documento UIML

La figura 8 muestra cómo el documento UIML es manejado internamente por cada uno de los componentes de Intergram.

Diagram: Crea el documento con la descripción de la relación entre los diferentes Objetos Interactivos.
Evalgram: Lee el fichero UIML para crear el StoryBoard, pudiendo añadirse descripciones respecto a la presentación (colores, posición, añadir elementos estáticos como textos, imágenes, etc.)
Logram: Se alimenta de la descripción UIML para conocer la estructura de navegación y sacar conclusiones en base a los logs del servidor.

Queda fuera del objetivo de este artículo mostrar todas las opciones que ofrece UIML, tan sólo mostrar su potencial para describir interfaces de usuario complejas, y que gracias a su expresividad, facilita la gestión de particularidades de los escenarios webs, como los menús de navegación, conjuntos de páginas que no cambian de estructura pero sí de contenido, etc.

Comentamos a continuación las ventajas de su utilización:

- Independencia del modelo de usuario
- La descripción intermedia del diseño escrita en UIML no está sujeta a los modelos de diseño escogidos; puede ser traducida desde modelos de análisis de tareas y navegación varios.
- Diagram, la herramienta que utiliza el diseñador para elaborar estos modelos, aún sujeta a ellos por su variada semántica, debe situarse a un nivel conceptualmente superior para abstraer al diseñador de esta pesada tarea. Internamente, la conversión será a UIML.
- Independencia del tratamiento gráfico de los elementos de la interfaz. Tras un prototipado y evaluaciones sucesivas con los usuarios finales, el fichero descripto en UIML queda fijado y puede pasarse como diseño final basado en el último prototipo presentado y evaluado correctamente por los usuarios.
- Reutilización: La permanencia de una descripción textual del diseño, permite acceder a el para realizar las modificaciones desde EVALGRAM.
- Por su formato textual y organizado facilita la recogida y almacenamiento de información en varias fases del desarrollo.

7. Conclusiones

El modelo propuesto en INTERGRAM hereda las características exigibles en los modelos centrados en el usuario. Se distingue por la exigencia de un orden de actuación, la inclusión de una notación intermedia y la inmersión de todas las fases en un proceso de evaluación constante.

Intergram es una herramienta de diseño de interfaces que sigue el proceso descrito en INTERGRAM, facilita el diseño usable, adaptado a las necesidades de los usuarios reales mediante técnicas de evaluación continua. Tiene en cuenta modelos y técnicas de ingeniería software específicas para este propósito. La utilización de una descripción intermedia en XML aporta independencia y permite la reusabilidad. Las herramientas Diagram, Evalgram y Logram están en un punto avanzado de realización. Destacamos la utilización de la metodología propuesta en el desarrollo de la aplicación Intergram.

Referencias

[1] Abrams,M., UIML Specification, 2004
[2] Boehm , B “A Spiral Model for Software Development and Enhacement” Computer, Vol.21, nº5, May 1988
[3] Card, S., Moran, T., Newell, A., The Psychology of Human-Computer Interaction, Lawrence Erlbaum, Hillsdale, 1983.

[4] Carroll, J. M. Scenario-Based Design, in Handbook of Human-Computer Interaction. Second Edition. M. Helander, T. Landauer and P. Prabhu. North-Holland, 1997

[5] Constantine, L. L., and Lockwood, L. A. D. Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design. Addison-Wesley, New York, NY, C. 1999

[6] Dix,A., J.Finlay, G. D. Abowd, R. Beale Human-Computer Interaction. Third Edition. Pearson Prentice Hall, 2004

[7] Ferre, X.; Juristo, N. ; Moreno, A.M., "Improving Software Engineering Practice with HCI Aspects", Computer Science, Vol.3026, pp. 349-363, 2004

[8] Greenberg, S. Teaching human-computer interaction to programmers. *Interactions*, ACM-Press, vol.3, nº 4, 62-76, 1996

[9] Hartson, R., Gray P., Temporal Aspects of Tasks in the User Action Notation", *Human Computer Interaction*, Vol.7, pp. 1-45, 1992

[10] Hix, D., and Hartson, H.R. Developing User Interfaces: Ensuring Usability Through Product and Process John Wiley and Sons, New York NY, 1993

[11] ISO (International Organization for Standardisation), 13407, p.7, Human-Centered Design Processes for Interactive Systems, 1997

[12] ISO, (International Organization for Standardisation), 13407, p.10, Human-Centered Design Processes for Interactive Systems, 1997

[13] Jackson, M., "Principles of Program Design". Academic Press, 1975

[14] Lorés, J.; Granollers, T., "La Ingeniería de la Usabilidad aplicada al diseño y desarrollo de sitios web", Capítulo del libro "Tendencias Actuales en la interacción Persona Ordenador: Accesibilidad, Adaptabilidad y Nuevos Paradigmas" (ISBN: 84-921873-1-X), pp. 119-144, Ediciones de la Universidad Castilla-La Mancha (Albacete, España) ; XIII Escuela de Verano de Informática, 2003

[15] Losada, B.; Lopistéguy, P.; Dagorret, P., "Etude de la Conception d'Applications Hypermédias" Congrès INFORSID'97, pp 133-146, Toulouse 1997

[16] Lozano, M.D, González, P, Ramos, I., Montero, F., Pascual, J. "Desarrollo y generación de interfaces de usuario a partir de técnicas de análisis de tareas y casos de uso", Inteligencia Arificial. Revista Iberoamericana de I.A. Nº 16 pp.83-92, 2002

[17] Nielsen, J., "Usability engineering at a discount". In Salvendy, G., and Smith, M.J. (Eds.), Designing and Using Human-Computer Interfaces and Knowledge Based Systems, Elsevier Science Publishers, Amsterdam. 394-401. 1989

[18] Nielsen, J., Big paybacks from 'discount' usability engineering, IEEE Software 7, 3 (May), 107-108., 1990

[19] Nielsen. J. Usability Engineering AP Professional. Boston, MA, 1993

[20] Olsina, L. Phd "Metodología Cuantitativa para la evaluación y comparación de la calidad de sitios web" 1998

[21] Parnas, D., "On the use of transition diagrams in the design of a user interface for an interactive computer program". In Proceeding of the 1969 24th ACM Conference, pp. 379-385

[22] Paterno, F. Model-Based Design and Evaluation of Interactive Applications, Springer-Verlag London, 1999

[23] Phillips, D. How to make a V-Shaped spiral and viceversa, The Cutter IT Journal, 1997

[24] Preece, J., Rogers, Y., Sharp H.; Benyon D., Holland S.; Carey T., "Human-Computer Interaction", Addison-Wesley, 1994

[25] Rosson,M.B., J.m. Carroll. "Usability Engineering". Morgan Kaufmann Publishers, 2002

[26] Snyder, C. Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces. Morgan-Kaufmann, 2003

[27] Sommerville, I. "Software Engineering" 4th edn. Wokingham, England:Addison-Wesley, 1992

[28] Sommerville, I. "Software Engineering" 5th edn. Wokingham, England:Addison-Wesley, 1995

[29] Stapleton, J. "Business Focused Development" 2nd. Edn. DSDM Consortium, Addison-Wesley, 2002

[30] Stone,D. C. Jarrett, M. Woodroffe, S. Minocha. User Interface Design and Evaluation. Morgan Kaufmann publishers, 2005