Usabilidad con AOP: Comparación de Enfoques y Herramientas.

Sandra Casas[1], Natalia Trejo[1], Juan Enriquez[1] y Roberto Farias[1],
,

[1] GISP. Instituto de Tecnología Aplicada, Universidad Nacional de la Patagonia Austral,
Campus Universitario, Piloto Lero Riversa S/N,
9400 Río Gallego, Argentina
{scasas, ntrejo, jenriquez, rfarias}@unpa.edu.ar

Abstract. Este trabajo realiza un recorrido por distintas propuestas que usan técnicas de programación orientadas a aspectos, para la captura automática de datos para la evaluación de usabilidad. Se propone un conjunto inicial de criterios de análisis (técnicos y no técnicos) que luego son usados para plantear el estudio comparativo. El objetivo es orientar a los potenciales adoptantes de los enfoques en cuanto a la variación de las características, y a los investigadores de estas áreas respecto de que ejes requieren mayor profundización o abordaje.

Keywords: Usabilidad; Programación Orientada a Aspectos, Logging; AspectJ

1 Introducción

La evaluación de la usabilidad [1] consiste en realizar pruebas para obtener información que permitan observar e identificar debilidades relacionadas al uso de las aplicaciones de software. Las cuatro formas básicas de evaluación son: automática (se calculan las métricas durante la ejecución de la aplicación), empírica (la usabilidad es evaluada testeando la aplicación con usuarios reales), formal (se usan modelos formales y fórmulas para el cálculo de medidas de usabilidad), e informal (se basan en reglas generales y la habilidad y experiencia de los evaluadores). Este proceso de evaluación implica varias actividades en función del método empleado, pero en general se agrupan en 3 pasos básicos [2]: Captura de datos, Análisis y Crítica.

Cualquiera sea la forma de evaluación que se adopte o combinación de ellas, la recolección de los datos es una tarea fundamental. En lo que respecta a la captura de datos automatizada se ha caracterizado por usar archivos de logs. Sus ventajas son varias, como: a) soportan la ejecución de las pruebas en laboratorios de usabilidad o en el ambiente natural del usuario; b) son transparentes para los usuarios; c) permiten con facilidad obtener información de múltiples usuarios; d) permiten detectar errores comunes y actuales; etc. Los logs aportan gran cantidad de datos a los evaluadores de usabilidad, además de poder contrastar comportamientos entre diferentes usuarios. Sin embargo, tienen desventajas, al emplear los logs generados por servidores webs o sistemas operativos, se obtiene una gran cantidad de información que resulta

irrelevante para la evaluación de usabilidad, son incapaces de registrar información específica. Por otro lado, añadir en las aplicaciones el código que realiza el logging de las actividades / tareas de usabilidad que interesan al evaluador de usabilidad, implica invadir la estructura interna de las aplicaciones agregando cientos de líneas de código, cuya remoción posterior es muy compleja.

A partir de la aparición de las técnicas de separación de concerns [3], y en particular de la Programación Orientada a Aspectos (AOP) [4], es posible diseñar módulos de código (aspectos) que realicen el seguimiento y log sin invadir y/o alterar la estructura interna de los sistemas. La posibilidad de realizar la recolección de los datos en forma dinámica, transparente para el usuario y sin invadir el código, ha provocado interés y varios enfoques se han presentado que aplican AOP a la evaluación de usabilidad.

Este trabajo realiza un recorrido por las distintas propuestas que usan la AOP para la captura automática de datos para la evaluación de usabilidad. Se propone un conjunto inicial de criterios de análisis (técnicos y no técnicos) que son empleados en el estudio comparativo. El objetivo es orientar a los potenciales adoptantes de los enfoques en cuanto a la variación de las características, y a los investigadores de estas áreas respecto de los ejes que requieren mayor profundización o abordaje.

Este trabajo se estructura de la siguiente manera, en la Sección 2 se presenta brevemente los conceptos básicos de la AOP, en la Sección 3 se presenta un resumen de los enfoques que usan AOP en la evaluación de la usabilidad, en la Sección 4 se presenta el conjunto de criterios de comparación, en la Sección 5 se presentan los resultados del análisis y comparación de los enfoques según los criterios escogidos y, en la Sección 6 se presenta la discusión y conclusiones.

2 Programación Orientada a Aspectos (AOP)

La AOP [4] es una técnica de programación que surgió hace una década y su objetivo es encapsular la funcionalidad transversal que genera código mezclado y diseminado, generalmente relacionado a requerimientos no funcionales, técnicos y/o de calidad. Ejemplos típicos de funcionalidad transversal son logging, traza, seguridad y persistencia. Con el fin de encapsular e implementar este tipo de funcionalidad la AOP introduce cuatro nuevas abstracciones: join-point, pointcut, advice y aspectos:

- Un join-point es un punto bien definido en la ejecución de un programa, por ejemplo la invocación a un método.
- Un pointcut es una expresión que especifica las condiciones de ejecución de un segmento de código (advice), es decir, determina el conjunto de join-points que serán interceptados por el aspecto, y expone algunos de los valores del contexto de los mismos.
- Un advice es un segmento de código similar a un método que se ejecuta en cada join-points especificado en los pointcuts.
- Un aspecto es un tipo transversal que encapsula pointcuts, advices y características transversales estáticas. Un aspecto es la unidad de modularización de la AOP.

Los aspectos se integran (componen) en el sistema por medio de un proceso especial llamado tejedor [5]. Actualmente hay extensiones de lenguajes

convencionales que soportan AOP para las aplicaciones implementadas con esos lenguajes de programación convencionales. AspectJ [6] es uno de los soportes AOP para Java.

3 Resumen de Enfoques

En esta Sección se describen brevemente los trabajos presentados que usan AOP en la evaluación de la usabilidad en los últimos años.

A. Framework de Diseño

Tarta y Moldovan [7] proponen un diseño de aspectos que soporta la evaluación automática de la usabilidad en aplicaciones de escritorio. Se plantea que mediante una jerarquía de aspectos se pueden reusar pointcuts que compartan los mismos puntos de entrada y salida (join-points). Los aspectos derivados definen nuevos pointcuts para la intercepción de errores, completitud de tareas, captura de pantallas y adquisición de datos por medio de cuestionarios. Los aspectos concretos por medio de los advices realizan el log de los datos. La estructura es específica de cada aplicación en la que se use. Se presentan simples diagramas y código de ejemplo en AspectJ.

Shekh y Tyerman [8] presentan el desarrollo de un framework AOP para la evaluación de la usabilidad con AOP. Las autoras registran los eventos de IU, por ejemplo los eventos del mouse. Respecto de las características del framework y diseño de aspectos no se brindan detalles, pero se indica que han seguido las recomendaciones de [7]. El framework esta desarrollado en AspectJ. Se presentan resultados de experimentos controlados y pautados en un laboratorio sobre la aplicación software ACIE Player.

B. Trayectoria de Eventos de Interface

Holzinger y otros [9] proponen la implementación de aspectos para seguir la trayectoria de algunos eventos de interface (entrada del teclado, menues y acciones arrastrar y soltar). Se plantea usar OBjective-C, agregando los aspectos a la jerarquía de objetos usando la tecnología "method swizzling", y extendiendo de clases. El diseño se centra en un aspecto que realiza el logging de manera centralizada y tres aspectos son necesarios para identificar los eventos de interface mencionados.

C. Generación automática de aspectos

Kronbauer y otros [10] [11] presentan una metodología para generar automáticamente los aspectos que realizan la captura de datos de usabilidad en aplicaciones móviles. La AOP se emplea en dos instancias y funciones bien definidas del enfoque. El código de los aspectos es generado mediante las especificaciones y acciones realizados en dos unidades (pasos) para lo que se ofrecen herramientas. La primer herramienta permite "la descripción automática de tareas", el ingeniero de usabilidad ejecuta esta herramienta en paralelo con la aplicación móvil a evaluar. Por medio de la ejecución en simultáneo se capturan las acciones de la interacción del ingeniero de usabilidad, que configuran una tarea determinada. Un aspecto va registrando los métodos en el camino de la tarea, los cuales se almacenan en un archivo XML. Además se definen

los parámetros de evaluación de usabilidad para dicha tarea. Luego con el archivo XML generado en el paso anterior (corresponde a la definición de una tarea), el ingeniero de usabilidad, a través de la herramienta "creación de aspectos de usabilidad" define que métricas se aplicarán en la evaluación, a partir de una biblioteca de métricas disponible. De esta segunda especificación se generan los aspectos para la evaluación de la usabilidad en AspectJ. La persistencia de los datos se trata en los aspectos. Los aspectos se usan para capturar datos para métricas de usabilidad convencionales (eficiencia, efectividad y satisfacción), también se puede recolectar información del contexto aportada por sensores (luminosidad-desplazamiento-posición) y de los perfiles de los usuarios (edad-sexo-estudios-etc.). Se presentan pruebas realizadas sobre aplicaciones móviles reales.

D. Toolkit para experimentos de usabilidad

Lettner y Holzmann [12] presentan un conjunto de herramientas para realizar experimentos relacionados al campo de la usabilidad sobre aplicaciones móviles. La propuesta consiste en posibilitar la conexión en forma transparente a los datos de interacción de usuario y calcular diversas métricas de bajo nivel durante los estudios de campo sin supervisión a los usuarios finales. El Toolkit se divide en tres partes principales: (i) el framework móvil para Android, (ii) un servidor back-end basado en Google App Engine y (iii) un servidor front-end basado en Google Web Toolkit.

La AOP se usa en el framework móvil. Como se describe en [13] los métodos del ciclo de vida de una aplicación Android, onCreate(), onResume(), onPause() y onDestroy() son interceptados por los aspectos para capturar el comportamiento de las actividades (Activity). Los datos se registran a nivel actividad y refieren esencialmente al comportamiento de navegación. Las métricas de bajo nivel que se recogen corresponden a errores de navegación, tiempo de sesión, llamadas a pantallas, clicks de botones, etc., las cuales son segmentadas y agregadas por grupo usuarios o usuarios individuales. Estas métricas se combinan con estadística para derivar métricas relacionadas a la usabilidad.

E. Trazas de Eventos

Traza de Eventos MVC

Yonglei [14] propone AOP para capturar automáticamente eventos de la interface de usuario en aplicaciones con arquitectura MVC (modelo-vista-controlador). Se propone una jerarquía de aspectos, en la cual se logra reusar solo el método que realiza el reporte (hora-fecha y evento ocurrido), los pointcuts y advices deben ser redefinidos en cada caso y aplicación, por ejemplo actualización de observer, manejadores y notificadores de eventos, cuadros de diálogo. Se presentan códigos de ejemplo en AspectJ y un caso de estudio simple.

Traza de Eventos IU

Un esquema muy similar al anterior, es el que presenta Yonglei [15], ya que este enfoque se basa en técnicas OA para ejecutar las trazas de los eventos de interface de usuario y recolectar información contextual, en aplicaciones de escritorio (WIMP). Propone una jerarquía de aspectos, la cual consiste básicamente en un método que contiene la lógica que permite hacer el reporte, los sub-aspectos definen los eventos a interceptar en los pointcuts y los advice que invocan al método que ejecuta el reporte. Se presentan simples códigos de ejemplo en AspectJ.

F. Instrumentación Interactiva
Bateman y otros [16] presentan el enfoque "Instrumentación Interactiva de Usabilidad" (IIU). Los evaluadores de usabilidad especifican qué acciones serán registradas (log) interactuando con los elementos de la interfaz de la aplicación objeto de la evaluación, eliminando la necesidad de apoyo adicional de programación. Esta primera actividad se apoya en AOP, y permite la instrumentación a través de la interacción directa con la aplicación a ser instrumentada; además de su simplicidad, especificación interactiva también proporciona un mecanismo natural para decidir qué elementos de un sistema están relacionados con tareas particulares y problemas de usabilidad. La herramienta UMARA da soporte al enfoque que ha sido probado en los softwares "PDF Split and Merge" y "MegaMek".

4. Criterios de Comparación

A continuación se presenta el conjunto de criterios utilizados en la próxima Sección para comparar los enfoques mencionados previamente. La compilación de este conjunto está centrada en la variabilidad de las diferentes características (técnicas y no técnicas). Como tal, el objetivo es de obtener un marco de clasificación y comparación flexible que ayude a los potenciales usuarios de los enfoques para seleccionar el más adecuado, y que ayude a otros investigadores a razonar respecto de las diferencias entre sus propios trabajos y las características existentes de otros. El conjunto de criterios permite contrastar las diferentes restricciones de cada enfoque e impone las fortalezas y debilidades a nivel de automatización y flexibilidad. Estos criterios forman un marco comparativo inicial que puede evolucionar siguiendo los nuevos desarrollos en los campos de usabilidad y/o AOP.

Plataforma. ¿A qué tipo de plataforma de software el enfoque está dirigido? Se distinguen tres tipos de plataforma: escritorio, móvil y web.
Nivel de abstracción. ¿A qué nivel de abstracción es posible realizar la captura de los datos de usabilidad? Se distinguen tres niveles: bajo, se interceptan eventos de manera general sin posibilidad de que sean acotados a una tarea del usuario específica, alto: los eventos observados se acotan a una tarea específica del usuario, y medio: algunos eventos pueden ser acotados a nivel tarea y otros no.
Soporte a métricas. ¿Los datos capturados aportan datos para calcular que clase de métrica? Usabilidad (efectividad, eficiencia-satisfacción), contexto (lectura de sensores o captura de pantallas), perfil de usuario (edad-sexo-formación-etc.), patrones (traza de eventos-navegación).
Uso de Aspectos. ¿Qué responsabilidad/función cumplen los aspectos en el enfoque? Se distinguen las siguientes:
-Identificación interactiva de tareas/elementos de IU. En un estadio previo a la recolección de datos, los aspectos se emplean para identificar y/o seleccionar los eventos que en ejecuciones posteriores serán observados.
- Traza de IU. Los aspectos interceptan determinados elementos de IU. Los elementos que suelen ser monitoreados son: eventos producidos por la acción del usuario como

presionar botones, movimiento de ratón, etc. y/o la ejecución de métodos de componentes gráficos como mostrar los cuadros de dialogo.
- Logging de datos. Esta actividad es realizada en los advices y/o métodos de los aspectos y consiste en registrar datos, que posteriormente se emplean para calcular métricas.
- Traza de tareas. Los aspectos por medio de pointcuts interceptan determinados puntos de ejecución (métodos) que configuran el inicio y fin de una tarea de usuario.
- Incorporación de cuestionarios: Los aspectos incorporan cuestionarios para obtener información subjetiva principalmente relacionada al factor satisfacción o perfil del usuario.
- Contexto: Los aspectos registran información relacionada al contexto (datos obtenidos de los sensores del dispositivo o captura de pantalla)

Aporte Empírico. ¿Cuál es la contribución empírica del enfoque? Una herramienta o un framework de diseño. ¿Cuál ha sido la validación empírica del enfoque? El enfoque se ha aplicado a sistemas reales o ejemplos pequeños.

Precondiciones. ¿Qué requisitos impone el enfoque para su aplicación y/o adopción? Conocimientos de la estructura interna de la aplicación cuya usabilidad será evaluada (diseño, arquitectura, implementación), conocimiento de AOP (conceptos y/o lenguaje OA), características de la aplicación a evaluar (lenguaje/arquitectura/ plataforma).

Usuarios Final. ¿Quién es el usuario final del enfoque? los desarrolladores de software o los ingenieros de usabilidad.

5. Resultados

Basados en los criterios introducidos en la Sección anterior comparamos los diferentes enfoques que usan AOP para capturar los datos para la evaluación de usabilidad. En la Tabla 1, se presenta el título abreviado del enfoque que será usado posteriormente.

Tabla 1.

Título Abreviado	**Identificador**
Framework de Diseño	A
Trayectoria de Eventos de Interface	B
Generación automática de aspectos	C
Toolkit para experimentos de usabilidad	D
Traza de Eventos MVC	E
Traza de Eventos IU	E
Instrumentación Interactiva	F

La Tabla 2 presenta una síntesis del análisis y comparación de los enfoques según todos los criterios definidos.

TABLA 2. ANALISIS Y COMPARACION DE ENFOQUES SEGUN LOS CRITERIOS PROPUESTOS

	Framework de Diseño (A)	**Trayectoria de Eventos de Interface (B)**	**Generación Automática de Aspectos (C)**	**Toolkit para experimentos de Usabilidad (D)**	**Traza de Eventos MVC (E)**	**Traza de Eventos IU (E)**	**Instrumentación Interactiva (F)**
Plataforma	Escritorio	Escritorio	Móvil	Móvil	Escritorio	Escritorio	Escritorio
Abstracción	Media	Baja	Alta	Media	Baja	Baja	Baja
Métricas	Usabilidad Contexto	Patrones	Usabilidad Contexto Perfiles de Usuario	Patrones	Patrones	Patrones	Patrones
Uso de Aspectos	Traza de IU Logging de datos	Traza de IU Logging de datos	Identificación interactiva de tareas y parametrización Traza de tareas Logging de datos Incorporación de cuestionarios Contexto	Traza de tareas(actividades) Logging de datos	Traza de elementos de IU Logging de datos	Traza de elementos de IU Logging de datos	Identificación interactiva de elementos de IU. Traza de elementos de IU Logging de datos
Propuesta	Jerarquía de Aspectos	Indicaciones de desarrollo y configuración.	Herramientas Automatic Task Description y Creation Usability Aspect	Toolkit - Framework	Jerarquía de Aspectos	Jerarquía de Aspectos Herramienta Prueba de conceptos	Herramienta UMARA
Validación Empírica	Dos aplicaciones de ejemplo	Una aplicación de ejemplo	Gingerbread Shuffle Mileage Cubed	ScotDroid	AccountManager (Deytel)	Una aplicación de ejemplo	PDF Split and Merge MegaMek
Precondiciones	Conocimiento del diseño /arquitectura de la aplicación Conocimientos de AOP Aplicaciones Java-AspectJ	Aplicaciones ObjectC Conocimiento del diseño /arquitectura de la aplicación Conocimientos de AOP	Aplicaciones Java (Android) –	Conocimiento del diseño/arquitectura de la aplicación Conocimientos de AOP Aplicaciones Java (Android)– AspectJ.	Conocimiento del diseño/arquitectura de la aplicación Conocimientos de AOP Aplicaciones Java-AspectJ Arquitectura MVC	Conocimiento del diseño/arquitectura de la aplicación Conocimientos de AOP Aplicaciones Java-AspectJ	Aplicaciones Java-AspectJ
Usuario Final	Desarrollador de Software	Desarrollador de Software	Ingeniero de Usabilidad	Desarrollador de Software	Desarrollador de Software	Desarrollador de Software	Ingeniero de Usabilidad

6. Discusión y Conclusiones

Trabajos Relacionados. Tarby y otros [17] comparan las AOP y los agentes interactivos para la evaluación temprana de la usabilidad de aplicaciones interactivas usando trazas. Ambos enfoques requieren ser incluidos en los estudios preliminares y de factibilidad del proyecto, especificación y codificación especifica. AOP no requiere diseño arquitectónico, a diferencias de los agentes dado que son parte de tales estructuras. Es decir AOP permite mantener intacto el código inicial.

Existen varios estudios comparativos y estados del arte, sobre los métodos de la evaluación de usabilidad, algunos incluso presentan taxonomías [2][18][19][20], la diferencia con este trabajo es que parten de alguna segmentación que en este trabajo forma parte de algún criterio de comparación. Por ejemplo, cuando se clasifican aplicaciones WIMP (escritorio) o cuando se analizan los métodos de captura de eventos. Asimismo no se toma como punto de partida la técnica de implementación como es el caso de este trabajo, que se focaliza en el uso de AOP.

Discusión. Los ejes de la discusión se plantean sobre los criterios de comparación y los aspectos más relevantes que del estudio surgen:

Plataforma. Llamativamente no se han encontrado en la literatura enfoques que usen AOP para la evaluación de la usabilidad de aplicaciones WEB.

Nivel de Abstracción. De los 7 trabajos analizados sólo uno proporciona medios para definir las tareas de usuario y asociar a estas los datos capturados. En los enfoques que han calificado con un nivel de abstracción “medio” se logra en forma restringida. Así el Framework de Diseño (A), define tareas solo para analizar ciertos eventos y el Toolkit para experimentos de usabilidad (D), se emplea el concepto de “actividad”, que en realidad esta mas asociado a las características estructurales de las aplicaciones Android, que a tareas de usuario. También se observa que los datos recolectados por aspectos en forma general (nivel de abstracción bajo), luego son de alguna forma asociados a tareas, como ocurre en Traza de Eventos MVC (E). La evaluación de la usabilidad requiere la posibilidad de vincular los datos/eventos obtenidos con la intención del usuario, y para ello es necesario contextualizarlos en el marco de una tarea. De otra forma, disminuye el significado y las posibilidades de la interpretación de los resultados. La traza/captura de eventos proporciona información de bajo nivel que puede ser usada en todo caso, para definir pruebas posteriores.

Soporte a métricas. Relacionada a la observación anterior, la usabilidad es débilmente evaluada ya que no es posible obtener datos para evaluar la efectividad, por ejemplo al no ser posible medir la completitud de las tareas. Por otro lado, el soporte para capturar y medir información contextual y de perfil de usuario, contribuye positivamente a comprender las acciones del usuario.

Uso de aspectos. El uso de aspectos se presenta como muy versátil, ya que pueden ser usados tanto para las funciones básicas de la captura de datos (traza y logging), y también pueden realizar otras funciones como posibilitar que el usuario identifique las tareas/eventos en forma interactiva o bien pueda calcular otras métricas.

Aporte empírico. Surge una diferencia muy notable al contrastar la validación empírica de los distintos aportes, lo que permite distinguir los diferentes niveles de madurez de las propuestas y posibilidades de adopción.

Precondiciones. Las precondiciones más fuertes están dadas para aquellos enfoques cuyo usuario final es el desarrollador de software, dado que debe conocer la aplicación a evaluar y AOP. En 6 de 7 casos se presenta una limitación importante respecto del lenguaje de la aplicación a evaluar (Java) ya que se ha utilizado AspectJ. AspectJ es una herramienta de propósito general, madura y consolidada en estos días, sin embargo es necesario abarcar otros lenguajes OA, para que los enfoques se puedan aplicar a software codificado en lenguajes que no sean Java.

Usuario Final. Los enfoques que proporcionan una herramienta que permite al Ingeniero de Usabilidad ser independiente para el desarrollo de las pruebas de usabilidad, en contra partida son menos flexibles, ante la necesidad de implementar cambios mínimos. Toda métrica y/o dato no soportado por la herramienta requiere que el generador sea modificado.

Conclusiones. En este trabajo se ha definido un conjunto de criterios de comparación de enfoques que usan AOP para la evaluación de la usabilidad. Estos pueden ser ampliados para completar la propuesta, también para plantear marcos de clasificación/categorización y/o para conformar un benchmark.

Los criterios de comparación se han aplicado a 7 enfoques, de este proceso emergen algunas conclusiones y ejes de trabajo futuro:

a) La captura de datos para la evaluación de la usabilidad con AOP en aplicaciones web es un tema cuyo tratamiento está pendiente.

b) Se deben abordar mecanismos para dar mayor y mejor soporte a la medición de datos/eventos asociados a tareas de usuario. Una hipótesis de las limitaciones actuales podría ser el uso de AspectJ, que es un lenguaje de propósito general, por lo que no dispone de mecanismos apropiados para el seguimiento de una tarea. Una posibilidad sería explorar el uso de lenguajes de aspectos de dominio específico (DSAL) que brinde soporte específico a la usabilidad.

c) La AOP es muy versátil en cuanto a las posibilidades que brinda a la evaluación de la usabilidad, ha demostrado capacidad para registrar diversos tipos de datos a diferentes clases de métricas, o bien ha posibilitado el proceso de identificación de tareas/eventos a evaluar. Los posibles usos de los aspectos no parecen estar agotados en estas propuestas y mucho menos aún el uso de otras técnicas de Separación de Concerns, como podría ser la aplicación de la programación orientada a features (FOP).

d) En cuanto al aporte empírico es necesario definir un benchmark común, basado en aplicaciones reales y de ser posible con usuarios reales, lo que facilitará el análisis de cada trabajo.

Referencias

1. ISO 9241-11. Ergonomic requirements for office work with visual display terminals. ISO, 1998.
2. Ivory, M. y Hearst, M., The State of the Art in Automating Usability Evaluation of User Interfaces. ACM Computing Surveys, vol. 33, no. 4, 2001.
3. Hürsch, W. y Lopes, C., Separation of Concerns. Northeastern University, TR NU-CCS-95-03, USA, 1995.

4. Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C. y Loingtier, J. Aspect-oriented Programming. Proceedings of the European Conference on Object-Oriented Programming (ECOOP), 1.997, LNCS 1241, Springer-Verlag
5. Piveta, E. y Zancanela, L., Aspect Weaving Strategies. Journal of Universal Computer Science, vol.9, no. 8, 2003.
6. Kiczales, G., Hilsdale, E., Hugunin, J., Kersten, M., Palm, L. y Griswold, W., An overview of AspectJ. Proceedings of the 15th European Conference on Object-Oriented Programming (ECOOP). 2001.
7. Tarta, A. y Moldovan, G., Automatic Usability Evaluation Using AOP. Automation, Quality and Testing, Robotics, IEEE International Conference, vol. 2, 2006.
8. Shekh A. y Tyerman, S., An Aspect-Oriented Framework for Event Capture and Usability Evaluation. Communications in Computer and Information Science: Evaluation of Novel Approaches to Software Engineering, vol. 69, Springer, pp. 107-119, 2010.
9. Holzinger, A., Brugger, M. y Slany, W., Applying Aspect Oriented Programming in Usability Engineering Processes - On the Example of Tracking Usage Information for Remote Usability Testing. Proceedings of the 8th. International Conference on electronic Business and Telecommunications. España pp. 53-56, 2001
10. Kronbauer, A. y Santos, C., Um modelo de avaliação da usabilidade baseado na captura automática de dados de interação do usuário em ambientes reais. Proceedings of the 10th Brazilian Symposium on on Human Factors in Computing Systems y 5th Latin American Conference on Human-Computer Interaction – pp. 114-123, 2011.
11. Kronbauer, A., Santos, C. y Vieira, V., Um Estudo Experimental de Avaliação da Experiência dos Usuários de Aplicativos Móveis a partir da Captura Automática dos Dados Contextuais e de Interação. Braziliam Symposium on Human Factors in Computing Systems (IHC). Brazil, 2012
12. Lettner, F. y Holzmann, C., Automated and unsupervised user interaction logging as basis for usability evaluation of mobile applications. 10th International Conference on Advances in Mobile Computing & Multimedia (MoMM), Indonesia, pp. 118-127, 2012.
13. Lettner F. y Holzmann, C., Sensing mobile phone interaction in the field. Proceedings of the 4th International Workshop on Sensor Networks and Ambient Intelligence, 2012.
14. Yonglei T., Automated Data Collection for Usability Evaluation in Early Stages of Application Development. 7th WSEAS in ACACOS 08, China.2008
15. Yonglei,T., Aspect-Oriented Instrumentation for Capturing Task-Based Event Traces. International Journal on Control System and Instrumentation, vol. 03, no. 01, 2012.
16. Bateman, S., Gutwin, C., Osgood, N. y McCalla, G., Interactive Usability Instrumentation. Symposium on Engineering Interactive Computing Systems, USA, 2009.
17. Tarby, J., Ezzedine, H., Rouillard, J., Tran, C., Laporte P. y Kolski, C., Traces Using Aspect Oriented Programming and Interactive Agent-Based Architecture for Early Usability Evaluation: Basic Principles and Comparison. HCI (1), Vol. 4550, pp. 632-641, doi:10.1007/978-3-540-73105-4_70, 2007.
18. Coutaz, J., Evaluation techniques: Exploring the intersection of HCI and software engineering. In R.N. Taylor and J. Coutaz, Eds., Software Engineering and Human-Computer Interaction, LNCS, pp. 35–48. Heidelberg, Germany: Springer-Verlag, 1995.
19. Whitefield A., Wilson F. y Dowell, J. A framework for human factors evaluation. Behaviour and Information Technology vol. 10, no. 1, pp. 65–7, 1991.
20. Balbo, S., Automatic evaluation of user interface usability: Dream or reality. Proceedings of the Queensland Computer-Human Interaction Symposium (Queensland, Australia, August). Bond University. 1995.