Universidad de las Ciencias Informáticas

Facultad 5

Método para evaluar la calidad de la arquitectura en cuanto a Requisitos No Funcionales en los software de Realidad Virtual

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autoras: Irene Suárez Morales

Elizabeth Rodríguez Castro

Tutora: Ing. Yusleidy Guelmes León

Ciudad de La Habana, Junio 2009

"El futuro de Cuba tiene que ser necesariamente un futuro de hombres de ciencia, de hombres de pensamiento"

Fidel

DATOS DE CONTACTO

Tutor

Ing. Yusleidy Guelmes León

Universidad de las Ciencias Informáticas, Ciudad de La Habana, Cuba.

Email: yguelmes@uci.cu

Tel: 2715

Apto: 115 105

AGRADECIMIENTOS

De Irene

A mi mamá, por ser el faro que guía mi vida, mami quiero ser como tú.

A mi papá, por todo su amor y dedicación, eres el mejor.

A mis abuelos Yaya y Papi, por malcriarme tanto.

A mi familia por su apoyo en todo momento.

A mi compañera de tesis Elizabeth, gracias por la gran paciencia y por ser una buena amiga.

A mis amistades, por estar conmigo en las buenas y en las malas.

A mis compañeros de aula, por hacerme la vida más fácil todos estos años.

A la tutora, por su apoyo.

A la Revolución.

Gracias a todos, los quiero.

De Elizabeth

A mi mamita y mi papito del alma, los seres que más quiero en este mundo, que nunca dejaron de estar ahí y confiaron ciegamente en mí.

A mi abuelita Flora, por ser tan buena conmigo.

A mis hermanos, por estar ahí siempre apoyándome, los quiero.

A mi amiga y hermana Yusleny, por estar siempre que la necesité dándome todo su apoyo, te quiero mucho.

A mi familia, por su confianza y apoyo.

A mi compañera de tesis Irene, por todo su apoyo.

A mis amigas Solainy, Liset, Anolandy, Yoslainy, Lisandra, gracias por estar ahí para mí.

A mis amistades Yanet, Yulie, Yisel, Lisandra, Yaimi, Papita, Elizeth, Yadira y Mirtha por todo el tiempo que hemos compartido juntas.

A todos mis compañeros de aula, por tantos momentos.

A Silvia Irene por toda su ayuda y ser una tutora más para nosotras.

A mi tutora, por su ayuda.

A Fidel Castro y a la Revolución por darme la oportunidad de ser una profesional.

A todos los quiero mucho.

DEDICATORIA

A mis abuelos Mama y Papa, se que estarían muy orgullosos de mi.

A mis padres, este logro es para ustedes.

Irene

A los mayores tesoros de mi vida, mis padres, mis hermanos y mis sobrinos.

Elizabeth

RESUMEN

La Facultad 5 de la Universidad de las Ciencias Informáticas se dedica al desarrollo de software de Realidad Virtual. En muchas ocasiones esta producción se ve afectada por la poca calidad que presenta el producto final, ya que al culminar la definición de la arquitectura no se verifica que ésta soporte los Requisitos No Funcionales. Es por ello que surge la necesidad de crear un método que permita la evaluación de la arquitectura en etapas tempranas del desarrollo del software, con el objetivo de elevar la calidad del producto final, registrando los puntos de riesgos o desventajas de determinadas decisiones arquitectónicas que se toman durante el análisis y diseño de la arquitectura.

PALABRAS CLAVES

Arquitectura de Software, Calidad de Software, Escenarios, Evaluación, Requisito de Software.

ÍNDICE

INTRODUCCIÓN

La calidad de los productos informáticos es una condición indispensable para que brinden utilidad, seguridad, durabilidad y puedan ser introducidos o aplicados de forma inmediata. Desarrolladores de software están constantemente bajo presión para entregar el código en tiempo y según lo presupuestado. Como resultado, muchos proyectos se centran en la entrega de funcionalidades descuidando la verificación de los Requisitos No Funcionales (RNF) tales como la fiabilidad, seguridad, mantenibilidad, portabilidad, exactitud, eficiencia, entre otros, trayendo como consecuencia que:

- ✓ El software no cumpla con la calidad necesaria.
- ✓ Falle en intentar soportar las necesidades del negocio.
- ✓ Exceso en el costo y tiempo de desarrollo.

De ahí la importancia de la evaluación rigurosa de los atributos de calidad del software antes de la implementación del software, pues mientras más tarde se detecten estos problemas, mayor será el impacto que tendrá en el costo de producción y menor la calidad, cualidad esencial a nivel mundial para poder ocupar un lugar en el amplio y competitivo mercado del software.

La Universidad de la Ciencias Informáticas (UCI) no se encuentra ajena a esta problemática, ya que muchos proyectos se han visto afectados sensiblemente en sus costos y cronogramas por problemas con la arquitectura. Los proyectos de Realidad Virtual (RV) que se desarrollan en la Facultad 5 de esta universidad son proyectos complejos de desarrollar, por lo general requieren de investigación y desarrollo de nuevos componentes o funcionalidades para satisfacer los requisitos del software. En muchos casos necesitan de alta precisión, el rendimiento en cuanto a velocidad de procesamiento y uso de memoria se convierten en aspectos claves a optimizar. La arquitectura de este tipo de sistemas debe tener en cuenta todos estos aspectos, sin pasar ninguno por alto. Los errores u omisiones en el diseño arquitectónico pueden destruir todo el trabajo de diseño e implementación logrado hasta el momento y elevar los costos del proyecto hasta el punto de hacerlo no factible o excesivamente costoso.

Tomando en cuenta que los proyectos de RV son de gran importancia económica y social para el país, se hace necesario enfrentar una serie de deficiencias para que los productos finales tengan la calidad necesaria. Las deficiencias existentes actualmente que motivaron esta investigación son:

- ✓ Por lo general estos proyectos cuentan con arquitectos de software que tienen poca o ninguna experiencia en este rol.
- ✓ Los RNF prácticamente no se documentan.
- ✓ No se verifica que la arquitectura que se diseña soporta todos los RNF antes de implementarla, lo cual puede conllevar a la detección de defectos críticos en etapas avanzadas del proyecto con innumerables costos e incluso el fracaso del proyecto.

Las irregularidades y deficiencias que se presentan en la concepción de la validación de los proyectos de RV inciden en que se prolongue el término de aplicación por la presencia de errores en la fase de desarrollo o prueba, disminuyendo la eficiencia del proceso de elaboración del producto final. Teniendo en cuenta que la UCI lucha por establecer una coherencia entre la producción, su inmediatez de aplicación y por alcanzar un lugar cimero en la economía, se declara como **problema a resolver**:

¿Cómo contribuir a la elevación de la calidad de los proyectos de RV que se desarrollan en la Facultad 5 de la UCI, a partir de la verificación de que la arquitectura soporte los RNF antes de su implementación?

El **objeto de estudio**: los procesos asociados a la verificación de los RNF en la arquitectura.

El **campo de acción:** los métodos para la verificación de los RNF en la arquitectura de los software de RV.

El **objetivo general** es proponer un método que valide la conformidad de la arquitectura con los RNF en los software de RV, contribuyendo a la detección de defectos técnicos del software tempranamente y a la elevación de la calidad del producto final.

Las **tareas** de investigación que se deben realizar para dar cumplimiento al objetivo general son:

- ✓ Comprobar la validez del problema para conocer la situación actual en los proyectos de RV.
- ✓ Realizar una sistematización teórica acerca de la concepción y aplicación de los métodos para evaluar los RNF.
- ✓ Diagnosticar el estado del documento Especificación de Requisito de Software en los proyectos de RV para identificar los problemas existentes.
- ✓ Elaborar la propuesta del método para validar la conformidad de la arquitectura con los RNF.
- ✓ Validar la propuesta del método.

La **idea a defender** es:

La aplicación del método propuesto para la verificación de los RNF en la arquitectura de los proyectos de RV, eleva la calidad de los mismos y evita la detección de defectos críticos en etapas avanzadas del proyecto.

Se emplearon en el trabajo los siguientes **métodos de investigación científica**:

- Métodos Teóricos

Analítico – Sintético: Se realiza un análisis de la bibliografía y se elabora una síntesis acerca de la misma.

- Métodos Empíricos

Observación: Se realizan visitas a los proyectos de RV para observar los procesos de verificación de los RNF en dichos proyectos, logrando comprender la situación problemática.

Entrevista: Se realizan entrevistas a los miembros del proyecto con el objetivo de precisar y comprobar la validez del problema.

Encuesta: Con el objetivo de determinar la importancia otorgada a los RNF y el costo que conlleva el logro de los mismos, se aplicó una encuesta a los líderes y arquitectos de los proyectos de RV.

El presente trabajo consta de 3 capítulos, que estarán estructurados de la siguiente forma:

Capítulo 1: **Fundamentación Teórica**, donde se plantean los elementos teóricos que sustentan el problema científico.

Capítulo 2: **Propuesta del Método,** se analiza la documentación de los proyectos de RV, las encuestas realizadas en los mismos y se propone el método para la evaluación de la calidad de la arquitectura en cuanto a RNF en los software de RV**.**

Capítulo 3: **Validación de la propuesta** seleccionando una muestra de especialistas intencional.

CAPÍTULO FUNDAMENTACIÓN TEÓRICA 1

1.1 Introducción

En el presente capítulo se abordan los fundamentos generales con las bases conceptuales que permiten el entendimiento del problema. Se realiza una descripción del objeto de estudio de la investigación al igual que la descripción del dominio y la situación problemática de la misma.

1.2 Requisitos del software

Según Kruchten, "el proceso de desarrollo de software parte de una necesidad expresada por el usuario u otro stakeholder. Estas necesidades son normalmente traducidas en los requerimientos[1]" [1]. Los requerimientos del software son una descripción abstracta de los servicios que el sistema debería proporcionar al cliente, y las limitaciones bajo las cuales éste debería operar [2].

Estos presentan tres importantes propósitos:

- ✓ Permiten a los desarrolladores entender cómo el cliente quiere que trabaje el sistema.
- ✓ Especifican a los diseñadores la funcionalidad y las características que el sistema debe tener.
- ✓ Especifican al grupo de prueba lo que deben demostrar para convencer al cliente que el sistema satisface sus necesidades.

Los requerimientos del software se clasifican en: Requerimientos Funcionales y Requerimientos No Funcionales según Bass, Clements, Kazman, Bosch, Dromey, Pressman, Sommerville [3; 4; 5; 6; 2] y la ISO 9126 [7], aunque categorizan los RNF de manera distinta, tanto unos como otros se corresponden con atributos de calidad deseables en un software. Según Whittem [8], "los requerimientos de software son documentados con cierto grado de rigurosidad y a través de un conjunto de especificaciones que permiten definir el alcance del desarrollo".

Estas especificaciones recogen no sólo la visión de los involucrados en el desarrollo sino también un conjunto de restricciones que impone el negocio en forma de reglas que limita las necesidades propias del

dominio. La especificación o análisis de estos requerimientos puede hacerse de manera formal (utilizando modelos matemáticos) o no formal (a través de modelos que representan las dimensiones de: datos, comportamiento y funcionalidad). De esta manera, las especificaciones de los requerimientos constituyen el punto de partida para lograr el acuerdo del equipo de desarrollo acerca del alcance del esfuerzo; y forman la base para la definición del modelo de diseño o la arquitectura del software, así como los casos de prueba correspondientes.

En la figura 1 se muestra la relación de estos conceptos de manera gráfica.

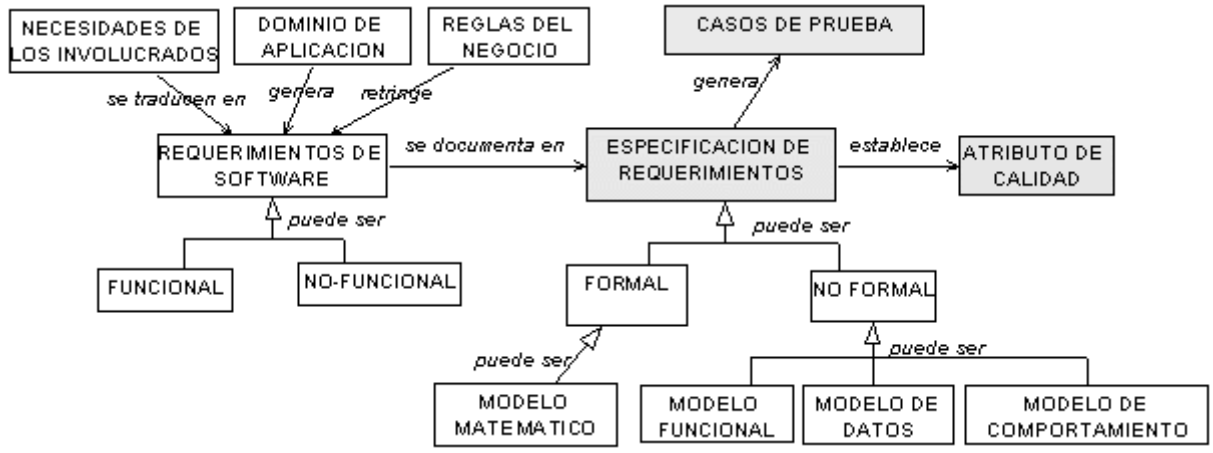


Figura 1: Relación de conceptos asociados.

1.2.1 Requisitos Funcionales

“Los Requisitos Funcionales describen lo que el sistema o el software deben hacer. Esta funcionalidad es la capacidad proporcionada y utilizable por uno o más componentes de un sistema, en ocasiones se le llaman a los Requisitos Funcionales conductuales u operacionales, ya que estos especificarán las entradas (estímulos) al sistema, los rendimientos (contestaciones) del sistema y las relaciones conductuales entre ellos” (Young, 2004).

1.2.2 Requisitos No Funcionales

"Los Requisitos No Funcionales especifican propiedades del sistema que el sistema debe cumplir, como restricciones del entorno o de la implementación, rendimiento, dependencia de la plataforma, facilidad de mantenimiento, extensibilidad, fiabilidad" (Jacobson, 2000).

Tienen que ver con las características del sistema, que incluye también interfaces de usuarios. Una de las clasificaciones más comunes que se asumen en proyectos productivos a nivel mundial está dada en aspectos como rendimiento/capacidad, seguridad, fiabilidad, portabilidad, mantenibilidad/ escalabilidad, reusabilidad, amigabilidad e interfaces. Según Wiegers, "los RNF son difíciles de expresar, los errores en el proceso de identificación y seguimiento de los RNF son muchas veces los más costosos de resolver".

Por lo que se plantea que los RNF se conocen como un conjunto de características de calidad, que es necesario tener en cuenta al diseñar e implementar un software, no modifican la funcionalidad del producto y constituyen de manera convincente parte de la razón fundamental del producto.

Este trabajo de investigación toma como referencia el modelo de calidad ISO/IEC 9126 que es un estándar internacional para la evaluación de la calidad de productos de software, el cual fue publicado en 1992 con el nombre de "Information technology –Software product evaluation: Quality characteristics and guidelines for their use". El estándar ISO-9126 [9] establece que cualquier componente de la calidad del software puede ser descrito en términos de una o más de seis características básicas, las cuales son: funcionalidad, confiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad; cada una de las cuales se detalla a través de un conjunto de subcaracterísticas que permiten profundizar en la evaluación de la calidad de productos de software.

Funcionalidad: La capacidad que tiene un producto de software para proveer funciones que satisfacen necesidades establecidas e implícitas, cuando el software es usado bajo condiciones específicas. Está compuesta por las siguientes subcaracterísticas:

- ✓ Apropiabilidad.
- ✓ Exactitud.
- ✓ Interoperabilidad.
- ✓ Cumplimiento con la funcionalidad.

Confiabilidad: La capacidad que tiene un producto de software para mantener su nivel de desempeño cuando éste es usado en condiciones específicas. Está compuesta por las siguientes subcaracterísticas:

- ✓ Madurez.
- ✓ Tolerancia a fallas.
- ✓ Recuperabilidad.
- ✓ Cumplimiento con la confiabilidad.

Usabilidad: La capacidad que tiene un producto de software para ser entendible, aprendido, utilizable y atractivo al usuario cuando éste es usado en condiciones específicas. Está compuesta por las siguientes subcaracterísticas:

- ✓ Comprensibilidad.
- ✓ Facilidad de aprendizaje.
- ✓ Operabilidad.
- ✓ Atractivo.
- ✓ Cumplimiento con la usabilidad.

Eficiencia: La capacidad que tiene un producto de software para proveer el desempeño apropiado relacionado a la cantidad de recursos usados, bajo condiciones determinadas. Está compuesta por las siguientes subcaracterísticas:

- ✓ Comportamiento en el tiempo.
- ✓ Utilización de recursos.
- ✓ Cumplimiento con la eficiencia.

Facilidad de Mantenimiento: La capacidad que tiene un producto de software para ser modificado. Modificaciones pueden incluir correcciones, mejoras o adaptación del software a los cambios de entorno, requisitos y especificaciones funcionales. Está compuesta por las siguientes subcaracterísticas:

- ✓ Trazabilidad.
- ✓ Facilidad de cambio.
- ✓ Estabilidad.

- ✓ Facilidad de ensayo.
- ✓ Cumplimiento con la facilidad de mantenimiento.

Portabilidad: La capacidad que tiene un producto de software para ser transferido de un ambiente a otro. Está compuesta por las siguientes subcaracterísticas:

- ✓ Adaptabilidad.
- ✓ Instalabilidad.
- ✓ Coexistencia.
- ✓ Reemplazabilidad.
- ✓ Cumplimiento con la portabilidad.

1.3 Calidad de software

En la actualidad hay cada vez mayor necesidad de realizar sistemas de información de calidad, ya que el desarrollo de los mismos aumenta cada vez más y la competencia en el mercado se hace más difícil. “La calidad del software es tan importante como la calidad de la productividad con la cual es generada. Esto es y será la llave del éxito junto con el servicio a los usuarios” [10].

1.3.1 Definiciones de Calidad de Software

“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” [11].

“El conjunto de características de una entidad que le confieren su aptitud para satisfacer las necesidades expresadas y las implícitas” [12].

“El conjunto de características de un producto que satisfacen las necesidades de los clientes y en consecuencia hacen satisfactorio el producto” [13].

Lograr el éxito en la producción de software es hacerlo con calidad y demostrarlo. Esto sólo es posible con la implantación de un Sistema para el Aseguramiento de la Calidad del Software directamente relacionado con la política establecida para su elaboración.

1.4 Aseguramiento de la calidad

Según Addison Wesley [14] “el aseguramiento de la calidad toma en cuenta todas aquellas acciones planificadas y sistemáticas necesarias para proporcionar la confianza en que un producto o servicio satisfaga los requisitos de calidad establecidos”. Es importante destacar que el aseguramiento de la calidad del software no se puede dar si no existe un control que garantice el cumplimiento de las fases o etapas que se han concretado previamente en una metodología, y sin una vigilancia permanente de todo el proceso de desarrollo, hechos que se pueden lograr con una continua evaluación de la calidad que se ha alcanzado en cada etapa del proceso antes de continuar adelante. Esta evaluación de la calidad permite realizar las rectificaciones necesarias a cualquier falla encontrada durante el proceso de desarrollo [15].

El aseguramiento de la calidad persigue varios objetivos como son:

- ✓ Revisar los productos.
- ✓ Revisar la documentación.
- ✓ Verificar el cumplimiento de las especificaciones.
- ✓ Verificar el cumplimiento de los requisitos.
- ✓ Conseguir un buen desempeño del Personal, Proceso y Producto.
- ✓ Evitar la necesidad de realizar cambios.
- ✓ Garantizar que el proceso se lleve de acuerdo con los estándares establecidos internacionalmente o internamente.

La presente investigación se basa en el Aseguramiento de la Calidad del Software, ya que procura garantizar la calidad de la arquitectura como medio para influir en la calidad del producto final.

1.5 Arquitectura de Software

La necesidad del manejo de la arquitectura de un sistema de software nace de los sistemas de mediana o gran envergadura, que se proponen como solución para un problema determinado. En la medida que los sistemas de software crecen en complejidad, bien sea por el número de requisitos o por el impacto de los mismos, se hace necesario establecer medios para el manejo de esta complejidad, sin la pérdida de tiempo o la elevación de los costos y gastos de operación. Con el tiempo se han ido desarrollando mecanismos para conseguir dichos propósitos.

En la última década cambió la visión que se tiene de los sistemas de software. Esta nueva visión se llamó “Arquitectura”. Desde los pequeños programas hasta los sistemas más grandes poseen una estructura y un comportamiento que los hace clasificables según su arquitectura. Se convirtió en el nuevo aspecto que hizo posible el estudio de sistemas ya implementados así como el desarrollo de nuevos, según la categoría del problema que resuelven y el tipo de Arquitectura de Software (AS) que se emplea para construirlos.

1.5.1 Definición de Arquitectura de Software

El significado de AS resulta, en general, bastante claro. Tal vez por ello, no existe una definición única y aceptada de la misma, distintos autores utilizan sus propias definiciones. Es muy habitual, citar varias de ellas de forma simultánea, con el fin de proporcionar una visión de conjunto.

“Una Arquitectura de Software es un conjunto de elementos arquitectónicos que tienen una determinada forma. Las propiedades restringen la elección de los elementos de la arquitectura mientras que la lógica captura la motivación de la elección de los elementos y la forma” (Perry y Wolf, 1992).

“Una Arquitectura de Software incluye la descripción de elementos a partir de los cuales se construyen los sistemas de software, interacciones entre esos elementos, patrones que guían la composición y restricciones sobre esos patrones. En general, un sistema de software particular se define en términos de una colección de componentes e interacciones entre dichos componentes. Tal sistema puede ser utilizado como un elemento en sistemas más grandes” (Garlan y Shaw, 1996).

“Una Arquitectura de Software de un programa o sistema de computación es la estructura o estructuras del sistema, el cual comprende componentes, las propiedades visibles externas de dichos componentes y las relaciones entre ellos” (Bass, Clements y Kazman, 1998).

“La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución” (IEEE 1471-2000).

La AS es considerada a grandes rasgos como la estructura del sistema, consistente en elementos y la relación entre ellos. Al describir la AS es importante identificar cuáles son los elementos de interés y seleccionar una manera apropiada para describirlos. Una Arquitectura del Software será estable en la

medida que garantice tanto los Requerimientos Funcionales del software como los Requerimientos No Funcionales [16].

Una AS ejerce una influencia notable sobre la calidad del sistema que se implementa, de ahí la importancia de evaluarla, para determinar si cumple con los requisitos de calidad exigidos.

1.5.2 Evaluación de la arquitectura

La arquitectura es un artefacto decisivo en la calidad del software que se desarrolla. Su evaluación permite:

- ✓ Mitigar los diferentes riesgos asociados con el desarrollo del software.
- ✓ Mejorar la visión de los procesos críticos y validar las decisiones de diseño que se tomaron.
- ✓ Tomar acciones tempranas.
- ✓ Valorar los RNF (Portabilidad, Facilidad de Mantenimiento, Eficiencia, Usabilidad, Confiabilidad, Funcionalidad) sin esperar a que el software se construya.

1.5.3 Objetivos de evaluar una arquitectura

La evaluación de una AS pretende medir propiedades del sistema en base a especificaciones abstractas como son los diseños arquitectónicos. Por ello, la intención es más bien la evaluación del potencial de la arquitectura diseñada para alcanzar los atributos de calidad requeridos. El propósito de realizar evaluaciones a la arquitectura, es fundamentalmente, para analizar e identificar riesgos potenciales en su estructura y sus propiedades que puedan afectar al sistema de software resultante, verificar que los RNF estén presentes en la arquitectura, así como determinar en qué grado se satisfacen los atributos de calidad.

(Kazman, 1996) propone un esquema general en relación a la evaluación de una arquitectura con respecto a sus atributos de calidad. En este sentido, Kazman y sus colegas afirman que de la evaluación de una arquitectura no se obtienen respuestas del tipo “si - no”, “bueno – malo” o “6.75 de 10”, sino que explica cuáles son los puntos de riesgos del diseño evaluado.

Una de las diferencias principales entre los planteamientos de Bosch y Kazman, es el enfoque que utilizan para efectos de la evaluación. El método de diseño de arquitecturas planteado por Bosch tiene como principal característica la evaluación explícita de los atributos de calidad de la arquitectura durante el

proceso de diseño de la misma. En este sentido, Bosch plantea las técnicas de evaluación basadas en: escenarios, simulación, modelos matemáticos y experiencia.

Por su parte, Kazman propone que resulta de poco interés la caracterización o medición de atributos de calidad en las fases tempranas del proceso de diseño, dado que estos parámetros son, por lo general, dependientes de la implementación. Su enfoque se orienta hacia la mitigación de riesgos, ubicando dónde un atributo de calidad de interés se ve afectado por decisiones arquitectónicas.

Tanto Bosch como Kazman indican la importancia de la especificación exhaustiva de los atributos de calidad como base para efectos de la evaluación de una AS. El punto es entonces definir los atributos de calidad en función de sus metas y su contexto y no como cantidades absolutas.

1.5.4 Cuándo evaluar una arquitectura

Es posible evaluarla en cualquier momento según (Kazman, 1996), pero propone dos variantes que agrupan dos etapas distintas: Temprana y Tarde.

Temprana: No es necesario que la arquitectura esté completamente especificada para efectuar la evaluación y que ésta abarque desde las fases tempranas del diseño y a lo largo del desarrollo.

Tarde: Cuando ésta se encuentra establecida y la implementación se ha completado. Este es el caso general que se presenta al momento de la adquisición de un sistema desarrollado.

(Bredemeyer D, 2002) propone que los requisitos arquitectónicos son necesarios para guiar las actividades de estructuración de un sistema de software. En el proceso de desarrollo se pueden aplicar diversos enfoques para garantizar el cumplimiento de los requisitos arquitectónicos, así como la evaluación de las alternativas presentadas. La evaluación provee indicadores que permiten, en las fases tempranas, la oportunidad de resolver problemas que pueden presentarse a nivel arquitectónico.

Independientemente de la metodología implementada, la intención es obtener una arquitectura con la documentación necesaria, y asegurar que el sistema cumple con los servicios y la funcionalidad que espera el usuario, además de los atributos de calidad asociados que deben cumplirse, y que dirigen las decisiones al momento de la construcción de la arquitectura del sistema (Bredemeyer D, 2002).

El tipo de evaluación en que se centrará la investigación será la evaluación temprana, o sea, en una fase donde predomina aún el diseño y no se ha comenzado a construir (implementar) el sistema a gran escala. De esta forma se pueden detectar riesgos técnicos y tomar decisiones que eviten el fracaso del proyecto.

1.5.5 ¿Quiénes participan en una evaluación?

Generalmente las evaluaciones a la arquitectura son llevadas a cabo por miembros del equipo de desarrollo, arquitecto, diseñador, analista, entre otros, contando con la presencia de especialistas de calidad de software que son quienes guiarán el proceso. También debe estar presente el cliente, pues en dependencia del resultado de la evaluación va a ser necesaria su opinión para realizar algunos cambios en la arquitectura o de ser necesario cancelar el proyecto.

1.5.6 Beneficios de realizar una evaluación arquitectónica

- ✓ Financieros.
- ✓ Exige una mejora a la documentación de la arquitectura.
- ✓ Mejora la arquitectura.
- ✓ Detección temprana de riesgos en la arquitectura.
- ✓ Permite la validación de requisitos y su priorización.
- ✓ Permite la documentación de las decisiones arquitectónicas tomadas y su fundamento.
- ✓ Permite constatar el grado de cumplimiento de una arquitectura con los RNF de un sistema.

1.5.7 ¿Qué resultados produce la evaluación de una arquitectura?

Una vez que sea efectuada la evaluación se debe elaborar un reporte formal donde se especifiquen todos los riesgos que se determinaron durante la evaluación, con la finalidad de que sean corregidos por los desarrolladores del software. El contenido del reporte responde a la siguiente pregunta: ¿Se ha diseñado la arquitectura más apropiada para el sistema?

1.5.8 Escenario

Un escenario consta de tres partes: el estímulo, el contexto y la respuesta. El estímulo es la parte del escenario que explica o describe lo que el involucrado en el desarrollo hace para iniciar la interacción con el sistema. Puede incluir ejecución de tareas, cambios en el sistema, ejecución de pruebas, configuración,

etc. El contexto describe qué sucede en el sistema al momento del estímulo. La respuesta describe, a través de la arquitectura, cómo debería responder el sistema ante el estímulo. Este último elemento es el que permite establecer cuál es el atributo de calidad asociado.

Los escenarios describen una utilización anticipada o deseada del sistema, y típicamente se expresan en una frase. Los mismos representan una abstracción de los requisitos más importantes de un sistema.

1.6 Métodos para evaluar la arquitectura

La AS es de suma importancia para el desarrollo de un sistema, ya que ella es el corazón de éste, impactando los niveles de calidad asociados al sistema. Por tal motivo, para garantizar la confiabilidad y calidad, los arquitectos y desarrolladores de software necesitan métodos que les den soporte durante el proceso de desarrollo. Es interesante destacar que la organización y descomposición de los atributos de calidad ha permitido el establecimiento de modelos específicos para efectos de la evaluación de la calidad arquitectónica.

1.6.1 Software Architecture Analysis Method (SAAM)

El método de evaluación SAAM evalúa distintos atributos de calidad como portabilidad, modificabilidad, escalabilidad, modularidad e integrabilidad. (Kazman, 1994). Este se enfoca en la enumeración de un conjunto de escenarios que representan los cambios probables a los que estará sometido el sistema en el futuro.

Como entrada principal, es necesaria alguna forma de descripción de la arquitectura a ser evaluada. Las salidas de la evaluación del método SAAM son las siguientes:

- ✓ Una proyección sobre la arquitectura de los escenarios que representan los cambios posibles ante los que puede estar expuesto el sistema.
- ✓ Entendimiento de la funcionalidad del sistema, e incluso una comparación de múltiples arquitecturas con respecto al nivel de funcionalidad que cada una soporta sin modificación.

1.6.2 Architecture Tradeoffs Analysis Method (ATAM)

El método de evaluación de arquitecturas ATAM está inspirado en tres áreas distintas: los estilos arquitectónicos, el análisis de atributos de calidad y el método de evaluación SAAM (Software Architecture Analysis Method). El nombre del método ATAM surge del hecho de que revela la forma en que una

arquitectura específica satisface ciertos atributos de calidad, y provee una visión de cómo los atributos de calidad interactúan con otros.

El método se concentra en la identificación de los estilos arquitectónicos o enfoques arquitectónicos utilizados. Estos elementos representan los medios empleados por la arquitectura para alcanzar los atributos de calidad, así como también permiten describir la forma en la que el sistema puede crecer, responder a cambios, e integrarse con otros sistemas, entre otros. Apoya a los involucrados con el proyecto a entender las consecuencias de las decisiones arquitectónicas respecto a los atributos de calidad (RNF como: seguridad, disponibilidad, facilidad de mantenimiento) del sistema.

1.6.3 Quality Attribute Workshops (QAW)

EL QAW proporciona un modo de identificar atributos de calidad importantes y clarificar exigencias de sistema antes de que la AS haya sido creada. QAW es un método para obtener y documentar explícitamente los atributos de calidad. También proporciona mecanismos para la captura de otros materiales arquitectónicamente relevantes. El QAW es una forma de descubrir, documentar y dar prioridad a un sistema de atributos de calidad al principio de su ciclo de vida.

1.6.4 Active Reviews for Intermediate Designs (ARID)

ARID es un método de bajo costo, conveniente para realizar la revisión de diseños en las etapas tempranas del proyecto. Según el autor, ARID es un híbrido entre Active Design Review (ADR) y Architecture Trade-Off Method (ATAM).

El método ARID involucra las partes interesadas para crear un conjunto de escenarios que se utilizan para "probar" el diseño de usabilidad y así determinar si el diseño puede ser utilizado por los ingenieros de software que deben trabajar con él. El ARID ayuda a encontrar problemas que dificultan la utilización con éxito de la concepción del diseño.

ADR es utilizado para la evaluación de diseños detallados de unidades del software como los componentes o módulos. Las preguntas giran en torno a la calidad y completitud de la documentación y la suficiencia, el ajuste y la conveniencia de los servicios que provee el diseño propuesto.

1.6.5 Cost-Benefit Analysis Method (CBAM)

El CBAM entrega bases para el análisis económico de los proyectos de desarrollo y sistemas basados en una evaluación a la arquitectura. Permite el análisis de los costos, beneficios, implicaciones y el calendario de las decisiones arquitectónicas. Asimismo, evalúa la incertidumbre en torno a las sentencias de los costos y beneficios, a fin de proporcionar una base para la toma de decisiones sobre el diseño de la arquitectura. Además posibilita maximizar las ventajas y reducir los gastos. El CBAM se basa en el método de evaluación arquitectónica ATAM. El mismo consta de 6 pasos, cada uno se pueden ejecutar en la primera y segunda etapa:

- ✓ Elegir los escenarios y estrategias de arquitectura.
- ✓ Evaluar los beneficios de control de calidad.
- ✓ Cuantificar los beneficios de la arquitectura.
- ✓ Cuantificar los costos y el calendario de arquitectura.
- ✓ Calcular la conveniencia.
- ✓ Tomar decisiones.

1.6.6 Attribute-Driven Design Method (ADD)

El ADD es un método de descomposición recursiva. Apoya la construcción de la arquitectura basando el proceso de diseño en los atributos de calidad que deben ser cumplidos, especialmente para los atributos de rendimiento, modificabilidad, la seguridad, fiabilidad / disponibilidad y facilidad de uso.

1.6.7 Método de Bosch

El método propuesto por (Bosch, 2000) plantea, que el proceso de evaluación debe ser visto como una actividad iterativa, que forma parte del proceso de diseño, también iterativo. Una vez que la arquitectura es evaluada, pasa a una fase de transformación, asumiendo que no satisface todos los requisitos. Luego, la arquitectura transformada es evaluada de nuevo. El método tiene dentro de sus objetos de estudio los siguientes elementos: estilos arquitectónicos, vistas arquitectónicas, patrones arquitectónicos, patrones de diseño y patrones de idioma. Es aplicable luego que el diseño de la arquitectura ha sido establecido.

Este proceso de evaluación de Bosch se divide en dos etapas:

Etapa I

1. Selección de atributos de calidad.

2. Definición de los perfiles.

3. Selección de una técnica de evaluación.

Etapa II

4. Ejecución de la evaluación.

5. Obtención de resultados.

1.6.8 Losavio

Es un método para evaluar y comparar AS candidatas, que hace uso del modelo de especificación de atributos e calidad adaptado del modelo ISO/IEC 9126. La especificación de los atributos de calidad haciendo uso de un modelo basado en estándares internacionales ofrece una vista amplia y global de los atributos de calidad, tanto a usuarios como arquitectos del sistema, para efectos de la evaluación. El método es aplicable luego que el diseño de la arquitectura ha sido establecido.

1.6.9 Método de Evaluación para Arquitecturas Basadas en Componentes (MECABIC)

El MECABIC está inspirado en ATAM, aunque con el objetivo de facilitar su aplicación sobre las Arquitecturas de Software Basadas en Componentes (ASBC) se incluyó en algunos de sus pasos un enfoque de integración de elementos de diseño, inspirado en la construcción de partes arquitectónicas adoptado por el Architecture Based Design (ABD) (Bachmann F, 2000).

El objetivo principal de MECABIC es evaluar y analizar la calidad exigida por los usuarios sobre Arquitectura de Software Basadas en Componentes (ASBC). El método adapta diferentes elementos de algunos métodos de evaluación arquitectónica (ATAM, Bosch, ABD, ARID, Losavio) y establece un conjunto de pasos para determinar la calidad de software basado en componentes.

Además se propone un árbol de utilidad inicial basado en el modelo de calidad ISO-9126 instanciado para AS propuesto por Losavio (IEEE, 2000) . La adopción de este modelo por parte del MECABIC permite concentrarse en características que dependen exclusivamente de la arquitectura, y al ser un estándar facilita la correspondencia con características de calidad.

1.7 Conclusiones Parciales

Se realizó un estudio detallado de los métodos existentes para la evaluación de la arquitectura, donde se pudo determinar qué:

- ✓ El método ATAM evalúa con más profundidad en relación con otros métodos, fundamentalmente en lo que respecta a los atributos de calidad.
- ✓ El método ATAM tiene un conjunto de pasos, un equipo de evaluación y un conjunto de salidas mejor definidas y no presenta ninguna restricción con respecto a la característica de calidad a evaluar.
- ✓ El método SAAM debe utilizarse cuándo el atributo de calidad Modificabilidad es el de mayor interés.
- ✓ El método ARID evalúa mejor la Factibilidad de la arquitectura.
- ✓ El método Bosch posibilita que los arquitectos seleccionen los atributos de calidad a evaluar de acuerdo con la importancia que tengan sobre el sistema.
- ✓ MECABIC como está basado en ATAM tiene muchas características en común con el mismo, pero lo diferencia el objetivo para el cual fue diseñado, pues el mismo es aplicable para arquitecturas basadas en componentes, incluye orientaciones para generar y discutir escenarios de evaluación iniciales, y un conjunto de preguntas a partir de las cuáles permite estudiar las decisiones arquitectónicas consideradas.
- ✓ Los métodos estudiados se centran en la evaluación temprana de la arquitectura, específicamente después de la definición de la arquitectura, excepto el ARID que evalúa durante el diseño de la misma.

Es bueno destacar que un método de evaluación no es mejor que otro, sino que evalúa mejor, en ciertas condiciones, un atributo de calidad dado, por lo que en dependencia de las condiciones y lo que se desea evaluar, será el método de evaluación empleado, que no tiene que ser solo uno, se pueden adoptar varios métodos para una evaluación considerando de cada uno los aspectos que se consideren que se adecuan al proceso de evaluación que se está llevando a cabo.

Tomando en cuenta que los software de RV en varias ocasiones han tenido insuficiencias a la hora de liberar el producto y que actualmente no se consta en los mismos de un método que detecte estos riesgos tempranamente, se pretende elaborar un método para evaluar la arquitectura de estos software ajustándolo a las necesidades de los productos.

Para la elaboración del mismo se tomó en cuenta primeramente la necesidad de valorar las decisiones arquitectónicas que se toman durante la definición de la arquitectura y la forma en la que se podían evaluar para determinar en qué medida se corresponden estas decisiones con los atributos de calidad que deba poseer el producto final.

CAPÍTULO 2 PROPUESTA DEL MÉTODO

2.1 Introducción

En este capítulo se realiza un análisis del documento de Especificación de Requisitos del Software (ERS) de los proyectos de RV de la Facultad 5, se exponen los resultados de la encuesta aplicada a líderes y arquitectos de estos proyectos y se presenta la propuesta del método de evaluación de la calidad de la arquitectura.

La Facultad 5 consta actualmente de 14 proyectos de RV, de los cuales dos de ellos se excluyeron de la muestra porque son proyectos de diseño y animación que no requieren de los procesos tradicionales de Ingeniería de Software, los mismos son: Meñique y Escenarios Virtuales.

2.2 Análisis del documento Especificación de Requisitos del Software en los proyectos de Realidad Virtual de la Facultad 5

Con el objetivo de realizar un estudio profundo en el proceso de especificación de requisitos que permita ratificar los problemas existentes, primero se procederá a definir qué es RV y luego se realizará un análisis del estado actual en el que se encuentra el documento ERS, concretamente los RNF que son en los que se centra la investigación.

2.2.1 Realidad Virtual

La RV consiste en simular todas las posibles percepciones de una persona, como los gráficos para la vista, sonido, tacto e incluso sensaciones de aceleración o movimiento. Todas estas sensaciones diferentes deben ser presentadas al usuario de forma que se sienta inmerso en el universo generado por el ordenador, hasta el punto de dejar de percibir la realidad y ser engañado, sentirse transportado (al otro lado de la pantalla) como si de un universo nuevo se tratase.

Autores como Ericka Corrado Padilla, Julián J. Delgado y Salvador Castañeda plantean que la RV es una simulación tridimensional generada o asistida comúnmente por computadora de algún aspecto del mundo real o ficticio, en el cual el usuario tiene la sensación de pertenecer a ese ambiente sintético o interactuar

con él. La RV permite interactuar con mundos tridimensionales de una manera más natural, por ejemplo, un usuario puede realizar acciones dentro de un modelo virtual, desplazarse, moverse, caminar a través de él o levantar cosas, y de esta forma experimentar situaciones que se asemejan a las de la vida real.

Aunque se ha aplicado mayoritariamente al mundo de los videojuegos, existen ya aplicaciones en medicina que han permitido importantes avances en la simulación de intervenciones quirúrgicas, el entrenamiento en tierra de los pilotos para facilitar su aprendizaje sin tener que arriesgar sus vidas.

2.2.2 Análisis de la documentación de los Requisitos de Software en los proyectos de Realidad Virtual

Con el propósito de conocer la calidad del proceso de especificación de requisitos en los proyectos de RV, se realiza un estudio de la documentación de los mismos, para ello esta investigación se rige por la norma IEEE-STD-830, la cual utiliza el proyecto de Calidad de la Facultad 5 para normalizar la ERS.

La ERS son detalles que se debe tener en cuenta para documentar un software en particular, programa, o juego de programas que realizan ciertas funciones en un ambiente específico. Las ERS pueden escribirse por uno o más representantes del proveedor, uno o más representantes del cliente, o por ambos.

Los puntos a medir para la ERS son los siguientes:

Tabla de Contenidos

1. Introducción
1.1 Propósito
1.2 Alcance
1.3 Definiciones, siglas, y abreviaciones
1.4 Referencias
2. Descripción global
2.1 Perspectiva del producto
2.2 Funciones del producto
2.3 Características del usuario
2.4 Restricciones
3. Los requisitos específicos

Se realizó un estudio preliminar de los 12 proyectos que constituyen el 100% de la población de nuestra muestra y se pudo determinar lo siguiente:

- ✓ Los proyectos: Simulador de Tiro, Laboratorios de Realidad Aumentada y Laboratorios Virtuales, son muy recientes y no han comenzado con el levantamiento de requisitos, por lo que se decidió que no entren en el caso de estudio.
- ✓ El proyecto Simulador de Autos no se encuentra produciendo en estos momentos. Todo el personal que lo desarrollaba fue cambiado y el personal actual se encuentra en una fase inicial de preparación; además, no cuenta con documentación significativa aparte del código fuente. Por todo esto se decidió no incluirlo como parte de la muestra.

En resumen, se analizarán los 8 proyectos restantes, constituyendo esta muestra el 66.67% de la población estudiada.

Para el análisis de los documentos se verifica el cumplimiento de la correcta especificación de cada uno de los puntos que debe tener la plantilla, los cuales se encuentran explícitamente explicados en el (Anexo 1), los mismos deben ser:

1-La introducción de la ERS debe proporcionar una apreciación global

Debe contener las subdivisiones siguientes:

a) el Propósito.

b) el Alcance.

c) las Definiciones, siglas, y abreviaciones.

d) las Referencias.

e) la Apreciación global.

2-Descripción global

Esta sección de la ERS se debe describir los factores generales que afectan el producto y sus requisitos. Esta sección no declara los requisitos específicos, y consiste en:

a) la perspectiva del Producto.

b) las funciones del Producto.

c) las características del Usuario.

d) las restricciones.

3- Requisitos específicos

Esta sección la ERS debe contener todos los requisitos del software a un nivel de detalle suficiente para permitirles a los diseñadores diseñar un sistema para satisfacer esos requisitos, y a los auditores probar que el sistema satisface esos requisitos. A lo largo de esta sección, cada requisito declarado debe ser externamente perceptible por los usuarios, operadores u otros sistemas externos. Esta es la parte más grande y más importante de la ERS, donde los RNF deben estar descritos con claridad, ser posibles de probar y delinear la importancia del mismo.

Resultado del estudio realizado:

Luego de realizar un exhaustivo análisis de la documentación de los proyectos estudiados, se encontraron diversos problemas en estos. Los RNF no se documentan en la plantilla que se exige, se dejan de evidenciar puntos importantes como el propósito de la ERS y las referencias que se deben realizar en este. En la mayoría de los casos no se muestran la perspectiva del producto, las características del usuario y las restricciones que puede poseer. Se puede constatar además que a los requisitos no se les da prioridad a la hora de especificarlos, algunos son imposibles de probar y prácticamente no se detallan. (Anexo 2).

2.3 Análisis de la encuesta

Con el objetivo de determinar la importancia otorgada a los RNF y el costo o esfuerzo que conlleva el logro de los mismos, se aplicó una encuesta a los líderes y arquitectos de los once proyectos de RV que se encuentran funcionando actualmente, donde estos dieron una categoría de Alta (A), Media (M), y Baja (B) a cada requisito como se muestra en el (Anexo 3).

La encuesta se basa en dos aspectos:

- ✓ ¿Cuáles de los siguientes RNF son más importantes en el proyecto?
- ✓ ¿Cuáles de los siguientes RNF son más costosos de lograr en el proyecto?

Los resultados que arrojó la encuesta para cada atributo de calidad son los que se muestran en la Tabla 1 y 2, donde se reflejan las evaluaciones que dieron los encuestados, tanto en el criterio de costo como el de importancia. Ver (Anexo 4)

Atributos	Alta	%	Media	%	Baja	%
Funcionalidad	23	95.83	1	4.17	0	0
Confiabilidad	14	58.34	8	33.33	2	8.33
Usabilidad	15	62.5	7	29.17	2	8.33
Eficiencia	21	87.5	3	12.5	0	0
Facilidad de Mantenimiento	8	33.33	12	50	4	16.67
Portabilidad	9	37.5	8	33.33	7	29.17

Tabla 1: Resultado de la encuesta aplicada a los proyectos de RV en el criterio de importancia.

Atributos	Alta	%	Media	%	Baja	%
Funcionalidad	12	50	10	41.67	2	8.33
Confiabilidad	12	50	10	41.67	2	8.33
Usabilidad	8	33.33	9	37.5	7	29.17
Eficiencia	14	58.34	8	33.33	2	8.33
Facilidad de Mantenimiento	4	16.67	13	54.16	7	29.17
Portabilidad	8	33.33	5	20.84	11	45.83

Tabla 2: Resultado de la encuesta aplicada a los proyectos de RV en el criterio de costo.

Luego de un exhaustivo análisis del resultado de las encuestas, se llega a la conclusión que en los proyectos de RV de la Facultad 5 se le hace más costoso o se le da mayor importancia ha determinado atributos de calidad, en dependencia de los objetivos del proyecto a desarrollar. De manera general se les atribuye mayor importancia a la: Funcionalidad, Eficiencia, Confiabilidad y Usabilidad, y los que requieren mayor esfuerzo para lograrlos son: Funcionalidad, Eficiencia, Confiabilidad. Ver (Anexo 5)

2.4 Propuesta del Método para la evaluación de la calidad de la arquitectura en cuanto a Requisitos No Funcionales en los software de Realidad Virtual

“MECASRV”

Esta investigación presenta el logro de un estudio que proporciona un Método de Evaluación para arquitecturas en los proyectos de RV de la Facultad 5, MECASRV.

El objetivo principal es evaluar la arquitectura para analizar e identificar riesgos potenciales en su estructura y sus propiedades antes de su implementación, y así elevar la calidad del producto final. El método adapta diferentes elementos de algunos métodos de evaluación arquitectónica (ATAM, MECABIC, Bosch) y establece un conjunto de pasos para determinar la calidad de la arquitectura.

2.4.1 Condiciones mínimas para aplicar el método de evaluación:

Después que se concluye la definición de la arquitectura del software, el arquitecto debe informar al líder de su proyecto que ya la arquitectura está lista para ser evaluada, para que posteriormente este solicite la evaluación de la misma. Para llevar a cabo el proceso de evaluación el grupo de evaluadores debe verificar que la arquitectura posea un conjunto de condiciones mínimas que permitan evaluar satisfactoriamente.

El equipo de evaluadores verifica:

- ✓ Los requisitos o atributos de calidad han sido correctamente especificados de forma que sean medibles y verificables.
- ✓ La arquitectura ha sido documentada apropiadamente en términos de diagramas, modelos y documentos cumpliendo con las normas de calidad establecidas en el Proyecto.
- ✓ La arquitectura tiene un nivel de especificación que hace factible someterla al proceso de evaluación.

2.4.2 Entrada del método

- ✓ El documento arquitectura.
- ✓ El documento ERS.
- ✓ Documento Visión.
- ✓ Plan de Mitigación de Riesgos.

2.4.3 Salida del método

Una documentación detallada donde se especifican todos los elementos analizados durante el proceso de evaluación.

2.4.4 Equipo de evaluación

Se propone como equipo de evaluación tres grupos de trabajo, tal como se muestra en la Tabla 3.

Tomando en cuenta lo difícil que se torna en la UCI el contacto directo con los clientes del software se propone para la aplicación del método que estén presentes en la evaluación con carácter obligatorio:

- ✓ El arquitecto (s).
- ✓ El líder y el analista.
- ✓ Los evaluadores.

Grupos	Definición	Fases en las que participan
Arquitectos	Responsables de generar y documentar una arquitectura de software para el sistema estudiado. MECASRV propone que los arquitectos sean los mismos del proyecto al que se le realizará la evaluación.	Todas
Evaluadores	Integrado por personas con conocimiento de calidad y arquitectura de software, principalmente experiencia en arquitecturas para software de RV, los mismos guiarán el proceso de evaluación de la arquitectura.	Todas
Involucrados	Son las personas involucradas de alguna manera con el software a evaluar: programadores, usuarios, gerentes, entre otros.	1 y 3.

Tabla 3: Equipos de evaluación de MECASRV.

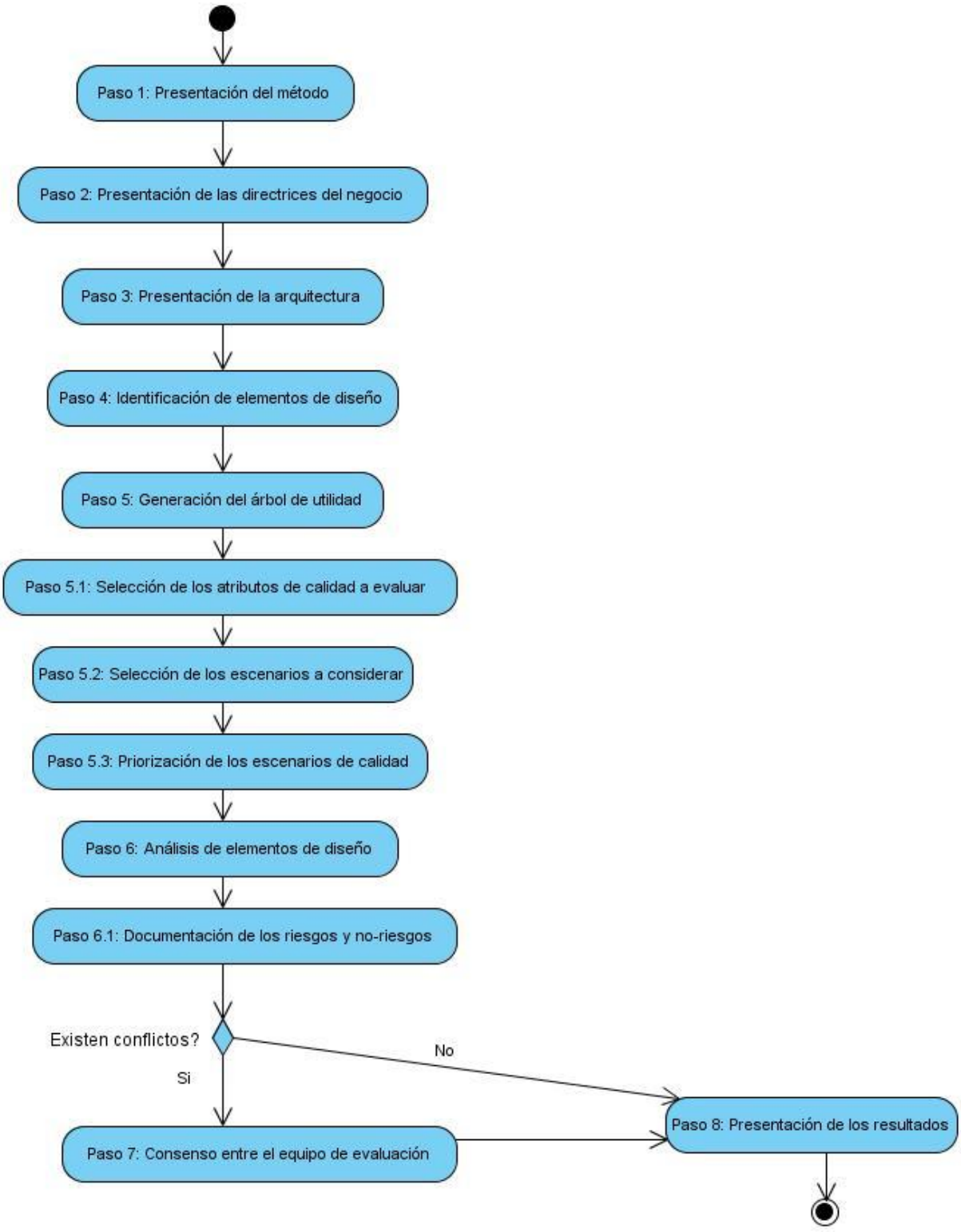
2.4.5 Instrumento y técnica de evaluación

Para la especificación de la calidad se hace uso de un árbol de utilidad, el cual tiene como nodo raíz la “bondad” o “utilidad” del sistema. En el segundo nivel del árbol se encuentran los atributos de calidad. Las

hojas del árbol de utilidad son escenarios, los cuales representan mecanismos mediante los cuales extensas (y ambiguas) declaraciones de cualidades son hechas específicas y posibles de evaluar. La generación del árbol de calidad incluye un paso que permite establecer prioridades. Para el análisis de la arquitectura se utiliza la técnica de escenarios, soportada por cuestionarios, para identificar lo que desean los participantes.

2.4.6 Fases del método

El método consta de 3 fases que a su vez contienen 8 pasos bien definidos que guían la aplicación del mismo, las cuales se muestran en el siguiente diagrama y se describen a continuación.


Paso 1: Presentación del método
Paso 2: Presentación de las directrices del negocio
Paso 3: Presentación de la arquitectura
Paso 4: Identificación de elementos de diseño
Paso 5: Generación del árbol de utilidad
Paso 5.1: Selección de los atributos de calidad a evaluar
Paso 5.2: Selección de los escenarios a considerar
Paso 5.3: Priorización de los escenarios de calidad
Paso 6: Análisis de elementos de diseño
Paso 6.1: Documentación de los riesgos y no-riesgos
Existen conflictos?
No
Si
Paso 7: Consenso entre el equipo de evaluación
Paso 8: Presentación de los resultados

Fase #1: Presentación

En esta fase se realiza una presentación del método MECASRV para que el equipo comprenda en qué punto de la aplicación del método se encuentran en cada momento. También se da a conocer la arquitectura inicial a evaluar y qué características de calidad se esperan lograr.

Paso 1. Presentación del método. En el primer paso el Jefe del equipo de evaluadores presenta el método ante los participantes del proyecto. Durante una reunión se explica la función que deberá cumplir cada persona implicada en el proceso de evaluación del proyecto. Este paso tiene como finalidad que todos los involucrados en el proceso de evaluación comprendan la secuencia de los pasos a seguir, los instrumentos utilizados en cada paso y las salidas del método.

Paso 2. Presentación de las directrices del negocio. El sistema a ser evaluado debe ser entendido por todos los participantes en la evaluación. En este paso, el líder o el analista del proyecto en evaluación presenta una visión general del sistema. El sistema debe ser presentado basándose en los siguientes aspectos:

- ✓ A quién va dirigido.
- ✓ Sus objetivos del negocio.
- ✓ Sus RF más importantes.
- ✓ Los atributos de calidad o RNF que se desean lograr.

Para esta fase de presentación el método propone la utilización de una plantilla, que en este caso sería una presentación Power Point con un aproximado de 12 diapositivas en las que se propone que se citen elementos esenciales tales como:

- Breve descripción del entorno en el que se desarrollará el sistema, a quién está dirigido el sistema, la necesidad actual y la forma en que el sistema va a satisfacer esas necesidades (3 a 4 diapositivas).
- Descripción de las limitaciones técnicas como son: la interoperabilidad con otros sistemas, si requiere de hardware o software, plataforma a utilizar, la reutilización de código, etc. (1 a 3 diapositivas).

- Atributos de calidad deseados (por ejemplo: funcionalidad, eficiencia, confiabilidad, facilidad de mantenimiento, usabilidad y portabilidad) y su importancia en el software (2 a 3 diapositivas).
- Glosario (1 diapositiva).

Plantilla 1 para el Paso 2: Presentación de las directrices del negocio.

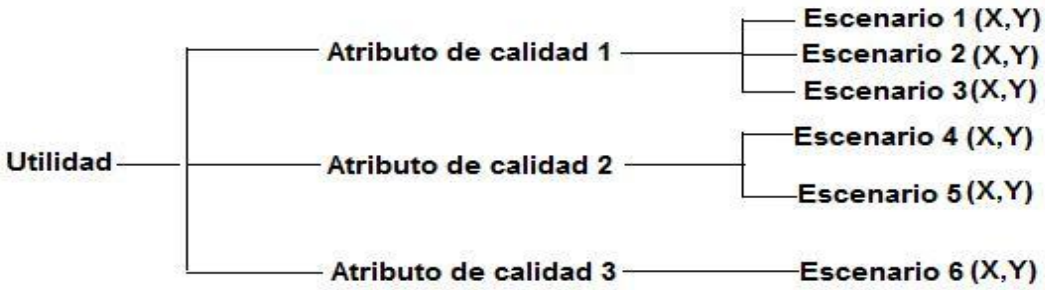
Paso 3. Presentación de la arquitectura. En este paso, el arquitecto debe explicar al equipo de evaluación cuál es la arquitectura a evaluar. En dicha presentación, el arquitecto debe cubrir restricciones técnicas (sistema operativo, hardware, otros sistemas con el cual el sistema debe interactuar) y los alcances arquitectónicos usados para obtener los requisitos. Debe transmitir lo esencial y de esta manera abrevia la información de la arquitectura que requiere el equipo.

Fase #2: Investigación y Análisis

En esta fase se identifican elementos de diseño que ayuden a analizar la arquitectura, se especifican los requisitos planteados durante el paso 2 y se analiza cómo responden los elementos de diseño considerados a los atributos de calidad.

Paso 4. Identificación de elementos de diseño. En este paso se contemplan alternativas de diseño de la arquitectura, que posteriormente serán analizadas para ver cómo responden a los atributos de calidad. La arquitectura debe ser evaluada completamente desde el sistema con sus componentes de mayor complejidad, hasta el mínimo nivel de complejidad que corresponde a los componentes, tomando en cuenta cada elemento arquitectónico como son las restricciones, patrones, estilo arquitectónico, software a utilizar, utilización de componentes para el desarrollo del software.

Paso 5. Generación del árbol de utilidad. El MECASRV proporciona un árbol de utilidad inicial, a partir del cual el arquitecto selecciona los atributos de calidad de mayor importancia para el sistema y para cada uno de ellos crea los escenarios que puedan poner en peligro el funcionamiento del software. Se recomienda que los arquitectos tengan en cuenta los atributos de calidad que presentan mayor importancia y son más difíciles de lograr en los proyectos de RV, los cuales son: Funcionalidad, Eficiencia, Confiabilidad y Usabilidad. Ejemplo de un árbol de utilidad.



A continuación se describen los pasos a seguir para obtener el árbol de utilidad.

Paso 5.1. El arquitecto o conjunto de arquitectos del proyecto selecciona los atributos de calidad de mayor importancia para el buen desempeño del sistema, que pueden ser todos los atributos de calidad o solamente los que estimen convenientes. Luego si el equipo de evaluadores determina que existe algún atributo que a su consideración debe ser evaluado y los arquitectos no lo han propuesto, plantean la inclusión del mismo y se llega a un consenso entre todo el equipo de evaluación para determinar si se evaluará este atributo. Como resultado se presenta el conjunto de atributos a ser evaluados.

Paso 5.2. En este paso el arquitecto presenta los diferentes escenarios a considerar al resto del equipo evaluador y se decide de manera conjunta si existe algún escenario que no fue contemplado por el arquitecto y que sea necesario incluirlo en el árbol de utilidad. Una vez obtenido los escenarios a incluir en el árbol de utilidad se procede a priorizarlos.

Paso 5.3. Priorización de los escenarios de calidad. En este paso el objetivo principal es ordenar los escenarios, ya que de esta manera los arquitectos cuentan con una mayor orientación para tomar decisiones arquitectónicas, y el equipo evaluador pueda estar más consciente de lo que esperan del sistema, y obtener una idea sobre en qué medida va a ser satisfecho. Para asignar un orden a los escenarios de calidad del sistema se utilizaran dos criterios fundamentales:

a) Evaluar el riesgo de no considerar el escenario. Se deben responder las preguntas: ¿qué pasa si este escenario no se cumple?, ¿es posible compensar el no responder a este escenario?

b) Evaluar el esfuerzo necesario para lograr cumplir con el escenario. En este caso se determina qué cambios o integraciones de nuevos componentes se deben realizar para satisfacer el escenario. Los escenarios más difíciles son aquellos en los que se debe consumir más tiempo de análisis.

Finalmente, se construye el árbol de utilidad con las salidas del paso anterior. La prioridad del escenario se define en este método como un par (X, Y) en el cual X define el esfuerzo de satisfacer el escenario, y la Y indica los riesgos que se corren al excluirlos del árbol de utilidad. Los posibles valores para X e Y son A (Alto), B (Bajo) y M (Medio). El árbol de utilidad generado se toma como un plan para el resto de la evaluación que realiza el método. Indica además al equipo evaluador dónde ocupar su tiempo y probar riesgos arquitectónicos.

Paso 6. Análisis de elementos de diseño. En este paso se analizan los elementos de diseño identificados en el paso 4 o nuevos elementos de diseño que puedan ser identificados, para determinar cómo influyen sobre la realización de los escenarios de calidad presentes en el árbol de utilidad generado en el paso 5.

Tomando en cuenta que en los proyectos de RV es muy frecuente la reutilización de componentes, ya que contribuye a minimizar el tiempo de desarrollo del proyecto, así como el esfuerzo empleado por parte del equipo de desarrollo, se hace necesario que se analicen las consecuencias que puede traer el uso de los mismos, para este caso se recomienda un conjunto de preguntas de análisis que facilitan la determinación de los posibles riesgos que pueda introducir el uso de dichos componentes, propuesto por el método de evaluación de arquitecturas basadas en componentes MECABIC, las mismas se muestran en la Tabla 4.

Considerando que un software de RV debe ser capaz de producir efectos visuales, auditivos y táctiles, coordinar estímulos que recibe el usuario con sus movimientos y acciones acordes al mundo simulado, el equipo de evaluación debe verificar que las herramientas y framework seleccionados cumplan con las necesidades que requiere el software y a su vez sean las más idóneas, para satisfacer los atributos de calidad que se evalúan.

Para el desarrollo de los software de RV se requiere de la implementación de un gran número de clases, por lo que se necesita el uso de patrones arquitectónicos que faciliten la organización y la asignación de responsabilidades dentro de las mismas, por lo que el equipo de evaluación deberá verificar si las decisiones que se tomaron fueron las más adecuadas, debido a que la no utilización de los patrones podría traer graves problemas en el atributo de calidad rendimiento.

Paso 6.1. Documentación de los riesgos y no-riesgos. Se deben estudiar los riesgos que introduce las decisiones arquitectónicas, o si ésta contribuye a algún riesgo ya determinado. También se pueden documentar decisiones que no han sido tomadas o asuntos no resueltos que a juicio del equipo de evaluación pueden ser resueltos en un futuro estudiando con más detalle el elemento de diseño considerado.

Atributos	**Preguntas de análisis.**
Funcionalidad	¿Puede la comunicación entre los componentes introducir imprecisiones en los servicios ofrecidos por los componentes? ¿Dónde se encuentran los componentes que permiten al sistema interactuar con otros sistemas?
Confiabilidad	¿Existen decisiones para minimizar el manejo incorrecto de datos en la comunicación entre componentes? ¿Cómo se detecta el funcionamiento incorrecto de un componente?
Eficiencia	¿Cómo es la relación entre el número de componentes de las diferentes partes de la arquitectura?
Facilidad de Mantenimiento	¿Cómo se verifica el funcionamiento correcto de un componente? ¿Cómo se verifica el estado de una comunicación entre componentes? ¿Cómo se facilita el reemplazo de un componente?
Portabilidad	¿Existe un mecanismo para determinar si todos los componentes del sistema se encuentran correctamente instalados? ¿Existe alguna manera de identificar las reglas para interactuar con componentes de sistemas externos a la arquitectura?

Tabla 4: Preguntas de análisis propuestas por MECABIC.

Fase #3: Resultados finales

En este momento se cuenta ya con un análisis de todas las decisiones que se han producido hasta el momento. Si existe una arquitectura adecuada para los RNF involucrados en la evaluación, entonces se puede pasar al paso 8 directamente. En caso contrario se continúa con el paso 7.

Paso 7. Consenso entre el equipo de evaluación. Si existen RNF que no han sido logrados, se debe brindar la posibilidad de que los stakeholder acepten replantear los atributos de calidad que no hayan podido ser alcanzados, para aprovechar posibles ventajas de la arquitectura. En caso de no estar presente el stakeholder será el analista del proyecto el responsable de esta tarea. Finalmente, se documentan todos aquellos RNF para los cuales no es posible encontrar una solución, justificando las razones.

Paso 8. Presentación de los resultados. Para finalizar la aplicación del método se presenta un resumen de la aplicación del método, de forma oral y escrita. Se deben incluir entonces, los siguientes documentos a partir de los cuales se inició la evaluación:

1) Contexto del negocio.

2) RNF.

3) Restricciones.

4) Análisis de elementos de diseño identificados.

5) Conjunto de escenarios.

6) Conjunto de preguntas para evaluación de atributos.

7) Árbol de Utilidad.

8) Riesgos.

9) No-riesgos.

10) Recomendaciones de los evaluadores.

2.5 Conclusiones Parciales

En este capítulo se materializó la propuesta del método para la evaluación de la calidad de la arquitectura de los software de RV, proporcionándole a la Facultad 5 un método con el que no se contaba y que contribuye a elevar la calidad de los productos finales que se desarrollan en cada uno de los proyectos. El método propuesto tiene como ventaja que evalúa no solo el cumplimiento de los RNF, sino que juzga las

decisiones arquitectónicas que se toman durante la definición de la arquitectura, logrando así una evaluación más exhaustiva de la misma, obteniendo de esta forma un número significativo de riesgos que deben tenerse en cuenta antes de la implementación del software.

CAPÍTULO 3
VALIDACIÓN DE LA PROPUESTA

3.1 Introducción

Como parte del ciclo de desarrollo del software, es importante verificar si los objetivos previstos se han cumplido y realmente la utilización de la propuesta eleva la calidad de los software de RV. Por lo que se debe recurrir a especialistas en los temas de calidad de software, AS y en el desarrollo de software de RV. Cada uno de los especialistas consultados, velará por la eficiencia de la propuesta, realizando una valoración de los aspectos que se definan para validar la misma.

Analizar los resultados de una propuesta propicia la realización de valoraciones acerca de la misma sobre la base de sus ventajas y limitaciones. Los resultados se consideran en virtud de su impacto y suele tomar como referencia opiniones tanto de personas con conocimientos del fenómeno como de otras que han interactuado con este.

Para la validación de la propuesta presentada en el Capítulo 2, se seleccionó una muestra de especialistas intencional. En el presente capítulo se hará una breve descripción de cómo se llevará a cabo el proceso de selección de los especialistas y los resultados obtenidos.

La aplicación de esta encuesta es de forma anónima, es decir, ningún especialista conoce la identidad del resto de los especialistas que componen el grupo de debate, para así evitar los efectos líderes. Se debe hacer una buena elección de los especialistas, para que el resultado sea el mejor.

3.2 Elección de los especialistas

Se entiende por especialista a una persona que posee una gran experiencia en el tema y es capaz de ofrecer valoraciones concluyentes del tema y hacer recomendaciones al respecto.

Para el procedimiento de la elección de los especialistas se debe llevar a cabo los siguientes pasos:

3.2.1 Determinar la cantidad de especialistas

Para la determinación de la cantidad de especialistas se recomienda que sean entre 7 y 30 especialistas siendo siempre la cantidad seleccionada impar.

Para la siguiente investigación se determinó la selección de 7 especialistas para la aplicación de la encuesta, los cuales son:

Especialista 1: Ing. Yusleidy Guelmes León.

Es graduada de Ing. Informática en el año 2005. Desde entonces se desempeña como profesora en la Facultad 5 de la UCI, con categoría Asistente, impartiendo fundamentalmente la asignatura Ingeniería de Software. Fue Asesora de Arquitectura y Tecnologías de la Facultad 5. Actualmente investiga en la temática de AS. Métodos de evaluación. Ha tutorado varias tesis de diploma vinculadas al proceso de desarrollo de software y relacionadas con temas de RV. Presenta cuatro publicaciones.

Especialista 2: Ing. Yurian Díaz Capote.

Es graduada de Ing. Informática en el año 2008. Es profesora de la Facultad 5 de la UCI y arquitecta principal y de información del Área Temática Videojuegos. Participó en UCIENCIA 2009 y está actualmente cursando posgrados.

Especialista 3: Ing. Jandrich Alfredo Domínguez Fortún:

Ingeniero Industrial. Graduado en Julio del 2005. Profesor de Matemática Aplicada. Especialista en Dirección General de Producción. Cursó el diplomado de Docencia Universitaria y de Dirección, entre otros cursos de postgrado. Máster en Gestión de Proyectos Informáticos. Ha participado en UCIENCIA 2006 y 2007, Fórum de Ciencia y Técnica 2006 y 2007, Informática 2007 y Octavo Congreso Nacional de Ciencia. Además presenta publicaciones como artículos científicos y memorias de eventos. Tiene la categoría de profesor asistente.

Especialista 4: Ing. Gerandys Hernández Casanova.

Ingeniero en Ciencias Informáticas, graduado en Julio del 2007. Profesor adiestrado. Fue líder del proyecto de Calidad de la Facultad 5 y subgerente del proyecto Fidetel, con Venezuela. En estos momentos se encuentra desempeñando el cargo de subgerente del proyecto Centro de Entrenamiento. Se encuentra cursando posgrados y haciendo la maestría en Gestión de Proyecto.

Especialista 5: Ing. José Manuel Pardo Matos.

Ingeniero en Ciencias Informáticas. Graduado en Julio del 2007. Profesor Adiestrado de la Facultad 5 de la UCI. Pertenece al Grupo de Auditoría y Revisiones Calisoft.UCI, en el cual es especialista con dos años de experiencia. Presenta publicaciones en temas como: Propuesta de Sistema para determinar el

aprovechamiento de la Jornada Laboral por medio del Muestreo del Trabajo. UCIENCIA 2007 y Guía metodológica para administrar Riesgos en Proyectos Productivos de RV. UCIENCIA 2008. Informática 2009. Ha cursado varios posgrados.

Especialista 6: Ing. Omar Correa Madrigal.

Ingeniero en Ciencias Informáticas. Graduado en Julio del 2007. Actualmente es líder del proyecto CNEURO, el cual pertenece al Polo de RV, teniendo 6 años de experiencia en dicho Polo. La categoría docente que ostenta es la de Instructor. Ha realizado 6 publicaciones.

Especialista 7: Ing. Dagoberto Marrero López.

Ingeniero en Ciencias Informáticas. Graduado en Julio de 2007. Profesor Adiestrado de la Facultad 5. Pertenece al Polo de RV hace 7 años. Dirigió el producto Energía para Aprender en el año 2008 y es actualmente el líder del proyecto DJKD. Participó en UCIENCIA 2007 con la presentación de la investigación Herramienta para el Manejo de Sonidos en Sistemas de RV.

3.2.2 Confirmar la participación de los candidatos

Una vez confirmado el listado de los especialistas se invitó personalmente a cada uno para participar en la evaluación. Al ser confirmada su participación, se estableció el listado final de los especialistas, informando a cada especialista su inclusión en el proceso a evaluar y las instrucciones necesarias para contestar las preguntas. De esta forma culmina el proceso de selección, logrando la participación de los especialistas seleccionados.

3.2.3 Elaboración de los cuestionarios

Para la elaboración de los cuestionarios se tuvo en cuenta los objetivos que se plantean lograr con el método propuesto.

El cuestionario realizado consta de 5 preguntas que permiten verificar la posibilidad real de aplicación de la propuesta realizada. Ver (Anexo 6).

Preguntas realizadas a los especialistas:

1- ¿Considera usted que el método propuesto para realizar la evaluación de la arquitectura se ajusta a las necesidades de los proyectos de Realidad Virtual?

2- ¿Cree usted que con la aplicación del método propuesto se logre incrementar la calidad del producto final?

3- ¿Considera usted que la evaluación de los atributos de calidad y las decisiones arquitectónicas son suficientes para el logro de los objetivos que propone el método?

4- ¿Considera usted que son suficientes los pasos propuestos para evaluar la arquitectura?

5- ¿Qué evaluación le otorgaría al método propuesto?

3.3 Resultados de la evaluación

Los resultados que arrojó la encuesta son los siguientes:

1- ¿Considera usted que el método propuesto para realizar la evaluación de la arquitectura se ajusta a las necesidades de los proyectos de Realidad Virtual?

Todos los especialistas consideran que el método propuesto para la evaluación de la calidad de las arquitecturas de los software de RV, se ajusta a las necesidades actuales de los proyectos de RV de la Facultad 5. Además el 100% de los encuestados le atribuyen gran importancia al trabajo, pues en ninguno de los proyectos del Polo de RV se lleva a cabo el proceso de evaluación de arquitecturas en cuanto a RNF.

2- ¿Cree usted que con la aplicación del método propuesto se logre incrementar la calidad del producto final?

Todos los especialistas (100 %) coinciden en el criterio de que si se lleva a cabo el método como se propone en la investigación traerá grandes beneficios a los proyectos de RV, incrementando así la calidad de los productos finales.

3- ¿Considera usted que la evaluación de los atributos de calidad y las decisiones arquitectónicas son suficientes para el logro de los objetivos que propone el método?

El 100 % de los especialistas coinciden en que la evaluación de los atributos de calidad y las decisiones arquitectónicas son suficientes para el logro de los objetivos que propone el método de evaluación de la calidad de las arquitecturas en los software de RV.

4- ¿Considera usted que son suficientes los pasos propuestos para evaluar la arquitectura?

El 100% de los especialistas consideran que con los pasos que plantea el método es suficiente para evaluar una arquitectura, aunque uno de los especialistas recomienda que las personas relacionadas con el desarrollo del software posean conocimiento de los criterios que se miden en el proceso de evaluación para que los tengan presente en el diseño de la arquitectura, y que se planifique en el cronograma del proyecto para tener en cuenta el tiempo que puede durar el proceso de evaluación.

5- ¿Qué evaluación le otorgaría al método propuesto?

La totalidad de los especialistas encuestados consideran que el método propuesto para la evaluación de la calidad de la arquitectura de los software de RV contribuye a elevar la calidad del producto final previendo así el fracaso del mismo y por tanto le otorgan la categoría de “Bien”.

Ver tabla resumen en el (Anexo 7).

3.4 Conclusiones Parciales

De manera general los especialistas en RV, AS y Calidad de Software que emitieron su criterio acerca de la investigación realizada, plantean que la propuesta se ajusta perfectamente a las necesidades que presentan actualmente los proyectos de RV. Además consideran que con las fases que propone el método se puede lograr un proceso de evaluación eficiente, logrando con el mismo que se eleve la calidad de los software de RV.

CONCLUSIONES

Al finalizar la presente investigación se dio cumplimiento al objetivo general concebido, con este se provee a la Facultad 5 de la UCI de un Método para evaluar la calidad de las arquitecturas en los software de RV, con el cual se logrará el incremento de la calidad final de los productos que se desarrollan en dichos proyectos, a partir de la detección de defectos críticos en etapas tempranas del software.

Para el desarrollo de la misma se realizó un estudio del arte, profundizando en el proceso de evaluación de arquitecturas; así como un análisis detallado de cómo se documentan los RNF en los proyectos de RV y la importancia que se le confiere a los mismos en dichos proyectos.

La propuesta se basa en algunos métodos como ATAM, MECABIC y Bosch, de los cuales se adaptaron los elementos que se consideraron que se adecuaban a las necesidades de los proyectos de RV y que de forma general enriquecen el proceso de evaluación.

Esta investigación aporta varios beneficios como son:

- ✓ Provee a la Facultad 5 de un método que permite evaluar la calidad de la arquitectura de los software de RV.
- ✓ Permite la detección de defectos críticos en etapas tempranas del desarrollo del software.
- ✓ Se elevará significativamente la calidad de los productos finales que se desarrollen en el Polo.

Con el propósito de determinar la aplicabilidad del método y verificar que cumpla con los objetivos para el cual fue creado, se realizó una encuesta a especialistas en los temas de Calidad del Software, AS y RV donde el 100% de la población encuestada calificó a la propuesta con la categoría de “Bien”, lo que significa que el método propuesto para la evaluación de la calidad de la arquitectura es aplicable a los proyectos de RV.

RECOMENDACIONES

Se recomienda:

- ✓ Poner en práctica el método propuesto en los proyectos de RV de la Facultad 5.
- ✓ Utilizar esta investigación como punto de partida y base de estudio para futuras investigaciones relacionadas con la evaluación de la calidad de las arquitecturas.
- ✓ Continuar perfeccionando el método para evaluar la arquitectura en los proyectos de RV, a través de la retroalimentación de su puesta en práctica.

REFERENCIA BIBLIOGRÁFICA

[1] **Kruchten, P.** The Rational Unified Process. Reading, MA: Addison Wesley Longman, Inc., 2003.

[2] **Sommerville, I.** Software Engineering. 7a. edición, 2005.

[3] **Bass, L., Clements, P. y Kazman, R.** Software Architecture in Practice. Third Edition: Addison Wesley, 2003.

[4] **Bosch, J.** Design & Use of Software Architectures: Addison-Wesley, 2000.

[5] **Dromey, G.** Cornering the Chimera. IEEE Software. pp. 33-43, 1996.

[6] **Pressman, R.** Ingeniería del Software: Un enfoque Práctico: McGraw Hill, 2004.

[7] ISO/IEC. Software Engineering – Software quality – General overview, reference models and guide to Software Product Quality Requirements and Evaluation (SQuaRE). 2002.

[8] **Whitten, J.; Bentley, L. y Dittman, K.** Systems Analysis and Design Methods. Sixth Edition: McGraw-Hill. 2004.

[9] **Sanders, Joc & Eugene Curran.** Software Quality. A Framework for Success in Software Development and Support: Addison Wesley.

[10] **Yourdon Edward,** The decline and Fall of the American Programmer, USA. 1995

[11] **R.S Pressman** 1992.

[12] ISO 8402 UNE 66-001-92.

[13] **Dr. Joseph M. Juran,** 1979.

[14] **Gonzales, J.** Las normas de la calidad del software: Addison Wesley. Iberoamericana. España, 2002.

[15] **Monsalve, L.** Calidad de los productos del software. España, 2000.

[16] **Kruchten, P.** The Rational Unified Process. Reading, MA: Addison Wesley Longman, Inc., 2003.

BIBLIOGRAFÍA

1- **Aristizábal Mejía, Nicolás.** Arquitectura de Software. 5 de agosto del 2008.

2- **Bass, Len, Klein, Mark y Bachmann, Felix.** Quality Attribute Design Primitives and the Attribute Driven Design Method.

3- **Borja, Andrés Felipe, Gómez Mojica, Johnny, De Villa, Raul Andres y Ramírez Madrid, Juan Pablo.** La importancia de la Arquitectura en el desarrollo de software de calidad. 17 de febrero del 2005.

4- **Bustacara Medina y Cesar Julio.** Evaluación de Arquitecturas de Software. 2008.

5- **Casanovas, Josep**. Usabilidad y arquitectura del software. 31 de septiembre del 2004.

6- **Cueva Lovelle, Juan Manuel.** Calidad del Software. Universidad de Oviedo. 1999.

7- **Cysneiros, Luis Marcio y Yu, Eric.** Non-Functional Requirements Elicitation. Universidad de Toronto.

8- **Darschy, Pablo y Snoeck, Michele.** Técnicas de Planificación Estratégica. Septiembre 2007.

9- **Díez González, Oscar.** Requisitos No Funcionales. 2006.

10- **García Fanjul, José y Álvarez, Claudio de la Riva.** Requisitos del software. Universidad de Oviedo. 2005.

11- **Grimán P, Anna C, Mendoza M, Luis E y Pérez de O, María A.** Criterios para la evaluación y selección de Técnicas de Evaluación Arquitectónica. Universidad Simón Bolivar.

12- **Grimán P, Anna, Mijares, Merizeh, González, Aleksander, Mendoza, Luis E y Pérez, María.** Análisis de Características (Modo Preselección) para Evaluar Plataformas de Componentes. Universidad Simón Bolívar.

13- **Hernán Astudillo.** Five ontological levels to describe and evaluate software architectures. 2005.

14- **Hernández León, Rolando Alfredo y Coello González, Sayda.** El Paradigma Cuantitativo de la Investigación Científica. UCI. Noviembre 2002.

15- **Icedo Ojeda, Rosa Virginia y Trejo Vargas, Jorge Moisés.** Software Architecture Assesment. Mayo 2003.

16- **Kingston, Gina.** Software Design Reviews Using the Software Architecture Analysis Method: A Case Study. Febrero 2000.

17- **Losavio, Francisca, Chirinos, Ledis, Lévy, Nicole y Ramdane-Cherif, Amar.** Características de calidad de Arquitectura de Software. 2003.

18- **Moreno García, María N, García Peñalvo, Francisco J y Polo Martín, María José.** Medición de la calidad del software en el ámbito de la especificación de requisitos. Universidad de Salamanca. 2000.

19- **Reynoso, Carlos Billy.** Introducción a la Arquitectura de Software. Universidad de Buenos Aires. Marzo 2004.

20- **Rodríguez, María Luisa y Garrido, José Luis.** Una aproximación a la evaluación de la calidad en el proceso de desarrollo de Sistemas Colaborativos. Universidad de Granada.

21- Software Engineering Institute http://www.sei.cmu.edu/publications

22- **Tekinerdogan, Bedir.** ASAAM: Aspectual Software Architecture Analysis Method. Ankara, Turkey.

23- **Vega Lebrún, Carlos, Rivera Prieto, Laura Susana y García Santillán, Arturo.** Mejores Prácticas para el establecimiento y aseguramiento de la calidad de software. 2008.

24- **Zamuriano Sotés, Roberto.** Aseguramiento de la calidad de software. Universidad del Valle, Bolivia.

ANEXOS

<u>Anexo 1</u>: Puntos que debe cumplir la planilla de Especificación de Requisitos del Software.

1-La introducción de la ERS debe proporcionar una apreciación global del documento ERS completo.

Debe contener las subdivisiones siguientes:

a) el Propósito.

b) el Alcance.

c) las Definiciones, siglas, y abreviaciones.

d) las Referencias.

a) Propósito

Esta subdivisión debe:

1) Delinear el propósito de la ERS.

2) Especificar a qué público intencional va dirigida la ERS.

b) Alcance

Esta subdivisión debe:

1) Identificar el producto (s) del software para ser diseñado por el nombre.

2) Explicar lo que el producto (s) del software hará y que no hará.

3) Describir la aplicación del software especificándose los beneficios pertinentes, objetivos y metas.

4) Ser consistente con las declaraciones similares en las especificaciones de niveles superiores (por ejemplo, las especificaciones de los requisitos del sistema), si ellos existen.

c) Definiciones, siglas, y abreviaciones

Esta subdivisión debe proporcionar las definiciones de todas las condiciones, las siglas, y abreviaciones que exigen interpretar la ERS propiamente. Esta información puede proporcionarse por la referencia a uno o más apéndices en el documento ERS o por la referencia a otros documentos.

d) Referencias

Esta subdivisión debe:

1) Proporcionar una lista completa de todas las referencias a los documentos que no se encuentran en la ERS.

2) Identificar cada documento por el título, número del reporte (si es aplicable), fecha, y publicación de la organización.

3) Especificar las fuentes de las referencias de donde se obtuvieron.

Esta información puede proporcionarse por la referencia a un apéndice o a otro documento.

2-Descripción global

Esta sección de la ERS debe describir los factores generales que afectan el producto y sus requisitos. Esta sección no declara los requisitos específicos, y consiste en:

a) la perspectiva del Producto.

b) las funciones del Producto.

c) las características del Usuario.

d) las restricciones.

a) Perspectiva del Producto

Esta subdivisión del documento ERS debe poner el producto en la perspectiva con otros productos relacionados. Si el producto es independiente y totalmente autónomo, debe declararse que así es. Si la ERS define un producto que es un componente de un sistema más grande, como frecuentemente ocurre, entonces esta subdivisión debe relacionar los requisitos de ese sistema más grande a la funcionalidad del software y debe identificar las interfaces entre ese sistema y el software.

b) Funciones del Producto

Esta subdivisión del documento ERS debe proporcionar un resumen de las funciones mayores que el software realizará.

c) Características del usuario

Esta subdivisión del documento ERS debe describir esas características generales de los usuarios intencionales del producto que incluye nivel educativo, experiencia, y la especialización técnica.

d) Restricciones

Esta subdivisión del documento ERS debe proporcionar una descripción general de cualquier otro punto que limitará las opciones de los diseñadores.

3**- Requisitos específicos**

Esta sección del documento ERS debe contener todos los requisitos del software a un nivel de detalle suficiente para permitirles a los diseñadores diseñar un sistema para satisfacer esos requisitos, y a los auditores a probar que el sistema satisface esos requisitos. A lo largo de esta sección, cada requisito declarado debe ser externamente perceptible por los usuarios, operadores u otros sistemas externos. Estos requisitos deben incluir por lo menos una descripción de cada entrada (el estímulo) en el sistema, cada salida (la contestación) del sistema, y todas las funciones realizadas por el sistema en la salida a una entrada o en el apoyo de la salida. Esta es la parte más grande y más importante del documento ERS, los principios siguientes aplican:

a) Deben declararse los requisitos específicos en conformidad con las características siguientes:

1) **Correcto**: una ERS es correcta si, y sólo si, cada requisito declarado se encuentra en el software. No hay ninguna herramienta o procedimiento que aseguren su exactitud. Alternativamente el cliente o el usuario pueden determinar si la ERS refleja las necesidades reales correctamente.

2) **Inequívoco**: una ERS es inequívoca si, y sólo si, cada requisito declarado tiene sólo una interpretación. Como mínimo, se requiere que cada característica de la última versión del producto se describa usando un único término. En casos donde un término en un contexto particular tenga significados múltiples, el término debe ser incluido en un glosario donde su significado sea más específico.

3) **Completo**: una ERS está completa si, y sólo si, incluye los elementos siguientes:

a) Los requisitos están relacionados a la funcionalidad, el desarrollo, las restricciones

del diseño, los atributos y las interfaces externas. En particular debe reconocerse cualquier requisito externo impuesto por una especificación del sistema y debe tratarse.

b) Están especificadas las respuestas que debe dar el software a los posibles datos de entrada del sistema.

c) Tiene todas las etiquetas llenas y referencias a todas las figuras, tablas, diagramas en la ERS y se encuentran definidas todas las condiciones y unidades de medida.

4) **Consistente**: una ERS es internamente consistente si, y sólo si, ningún subconjunto de requisitos individuales genera conflicto en él.

5) **Delinear que tiene importancia y/o estabilidad**: una ERS debe delinear la importancia y/o estabilidad si cada requisito en él tiene un identificador para indicar la importancia o estabilidad de ese requisito en particular.

6) **Comprobable**: Un requisito es comprobable si, y sólo si, existe algún proceso rentable finito con que una persona o la máquina puede verificar que el producto del software reúne el requisito. En general cualquier requisito ambiguo no es comprobable.

7) **Modificable: u**na ERS es modificable si, y sólo si, su estructura y estilo son tales que puede hacerse cualquier cambio a los requisitos fácilmente, completamente y de forma consistente mientras conserva la estructura y estilo.

8) **Identificable**: una ERS es identificable si el origen de cada uno de sus requisitos está claro y facilita las referencias de cada requisito en el desarrollo futuro o en la documentación del mismo.

b) Todos los requisitos deben ser singularmente identificables.

c) Debe prestarse la atención debida a la organización de los requisitos para aumentar al máximo la legibilidad.

Anexo 2: Resultado del análisis de la documentación de ERS de los proyectos de RV

Proyectos	1-Introducción	2-Descripción global	3-Requisitos específicos
Juegos CNEURO	a-si b-si c-si d-si	a-no b-si c-si d-no	Se encuentran bien especificados aunque no se prioriza ningún atributo.
Juegos de Consola	a-si b-si c-no d-no	a-si b-si c-si d-si	Se encuentran bien especificados.
Juegos ICAIC	a-si b-si c-si d-si	a-no b-si c-no d-si	No se encuentran correctamente especificados, no se prioriza ningún requisito.
Paseos Virtuales	a-no b-si c-si d-si	a-si b-si c-no d-no	Se encuentran bien especificados aunque no se prioriza ningún atributo.
Simulador Quirúrgico	a-no b-no c-no d-no	a-no b-si c-no d-no	Se encuentran bien especificados pero no se prioriza ningún atributo.
DJKD	a-si b-si c-si d-no	a-no b-si c-si d-no	Se encuentran bien especificados pero no se prioriza ningún atributo.
Herramienta de desarrollo de sistemas para RV	a-no b-si c-si d-no	a-no b-si c-no d-no	Se encuentran bien especificados.

Entrenadores **Aduana**	a-si b-si c-no d-si	a-no b-no c-no d-no	Se encuentran especificados, pero no denotan la prioridad entre los atributos.

<u>Anexo 3</u>: Encuesta aplicada a los líderes de proyectos y arquitectos

La encuesta se basa en dos aspectos fundamentales:

- ✓ ¿Cuáles de los siguientes RNF son más costosos de lograr en el proyecto?
- ✓ ¿Cuáles de los siguientes RNF son más importantes en el proyecto?

Evaluarlos dándole una categoría de Alta (A), Media (M), y Baja (B) a cada requisito.

Requisitos No Funcionales, basados en la Norma ISO/IECE 9126

Funcionalidad: La capacidad que tiene un producto de software para proveer funciones que satisfacen necesidades establecidas e implícitas, cuando el software es usado bajo condiciones específicas. Está compuesta por las siguientes subcaracterísticas:

- Apropiabilidad.
- Exactitud.
- Interoperabilidad.
- Cumplimiento con la funcionalidad.

Confiabilidad: La capacidad que tiene un producto de software para mantener su nivel de desempeño cuando éste es usado en condiciones específicas. Está compuesta por las siguientes subcaracterísticas:

- Madurez.
- Tolerancia a fallas.
- Recuperabilidad.
- Cumplimiento con la confiabilidad.

Usabilidad: La capacidad que tiene un producto de software para ser entendible, aprendido, utilizable y atractivo al usuario cuando éste es usado en condiciones específicas. Está compuesta por las siguientes subcaracterísticas:

- Comprensibilidad.
- Facilidad de aprendizaje.
- Operabilidad.

- Atractivo.
- Cumplimiento con la usabilidad.

Eficiencia: La capacidad que tiene un producto de software para proveer el desempeño apropiado relacionado a la cantidad de recursos usados, bajo condiciones determinadas. Está compuesta por las siguientes subcaracterísticas:

- Comportamiento en el tiempo.
- Utilización de recursos.
- Cumplimiento con la eficiencia.

Facilidad de Mantenimiento: La capacidad que tiene un producto de software para ser modificado. Modificaciones pueden incluir correcciones, mejoras o adaptación del software a los cambios de entorno, requisitos y especificaciones funcionales. Está compuesta por las siguientes subcaracterísticas:

- Facilidad de cambio.
- Estabilidad.
- Facilidad de ensayo.
- Cumplimiento con la facilidad de mantenimiento.
- Trazabilidad.

Portabilidad: La capacidad que tiene un producto de software para ser transferido de un ambiente a otro. Está compuesta por las siguientes subcaracterísticas:

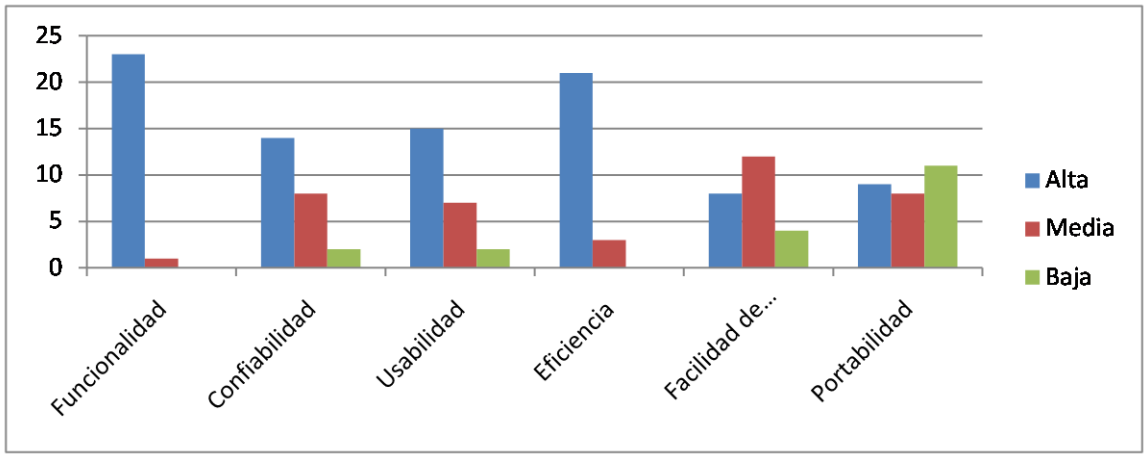
- Adaptabilidad.
- Reemplazabilidad.
- Cumplimiento con la portabilidad.
- Instalabilidad.
- Coexistencia.

Anexo 4: Resultado de la encuesta

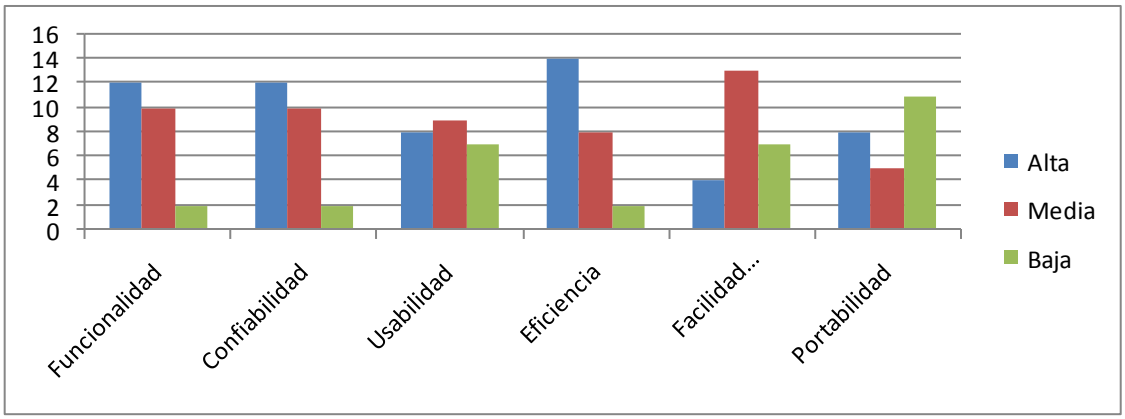
¿Cuáles de los siguientes RNF son más importantes en el proyecto?												
Proyectos	Requisitos No Funcionales											
	Funcionalidad		Confiabilidad		Usabilidad		Eficiencia		Facilidad de Mantenimiento		Portabilidad	
Juegos CNEURO	M	A	M	M	A	A	A	A	M	M	M	M
Juegos de Consola	A	A	A	A	A	A	A	A	A	A	M	A
Juegos ICAIC	A	A	M	M	M	A	A	A	B	M	M	A
Laboratorios Virtuales	A	A	A	B	M	A	A	A	M	M	M	M
Paseos Virtuales	A	A	A	M	A	A	A	M	M	A	A	A
Simulador Quirúrgico	A	A	A	A	M	B	A	A	B	M	B	B
Simulador de Tiro	A	A	M	A	M	B	A	A	M	M	B	B
Entrenadores Aduana	A	A	M	M	A	A	A	A	A	A	A	M
Compilación de Juegos	A	A	A	B	A	A	A	A	M	M	M	B
Simulador de Auto	A	A	A	A	M	M	M	M	B	B	B	B
Laboratorio de Realidad Aumentada	A	A	A	A	A	A	A	A	A	A	A	A
Herramienta de Desarrollo de Sistemas para Realidad Virtual	A	A	A	M	A	M	A	A	A	M	A	A

¿Cuales de los siguientes RNF son mas costosos de lograr en el proyecto?												
Proyectos	Requisitos No Funcionales											
	Funcionalidad		Confiabilidad		Usabilidad		Eficiencia		Facilidad de Mantenimiento		Portabilidad	
Juegos CNEURO	M	B	M	M	A	A	A	B	B	A	M	A
Juegos de Consola	M	B	A	B	B	B	A	M	M	M	B	M
Juegos ICAIC	M	A	A	M	B	M	M	A	B	M	M	A
Laboratorios Virtuales	A	A	A	A	B	B	M	A	M	M	A	B
Paseos Virtuales	A	A	M	A	M	A	M	A	M	M	M	A
Simulador Quirúrgico	A	A	A	A	A	B	A	M	B	M	B	A
Simulador de Tiro	A	M	M	A	M	B	M	M	B	B	B	B
Entrenadores Aduana	A	A	M	M	A	M	A	A	A	M	B	M
Compilacion de Juegos	M	M	M	B	M	M	A	A	M	A	B	A
Simulador de Auto	A	A	A	A	M	M	A	A	B	B	B	B
Herramienta de Desarrollo de Sistemas para Realidad Virtual	M	M	M	M	M	A	M	B	M	A	A	A
Laboratorio de Realidad Aumentada	M	M	A	A	A	A	A	A	M	M	B	B

Anexo 5: Gráficas de la encuesta



Gráfica 1: Resultado de la encuesta aplicada a los proyectos de RV en el criterio de importancia.



Gráfica 2: Resultado de la encuesta aplicada a los proyectos de RV en el criterio de costo.

Anexo 6: Encuesta realizada a los especialistas

Estimado compañero (a):

La presente encuesta tiene como objetivo validar el método propuesto para la evaluación de la calidad de las arquitecturas en los software de Realidad Virtual. Es por ello que solicitamos su colaboración a través de sus respuestas a las preguntas que aparecen a continuación, lo que será de gran utilidad para esta investigación. Le pedimos su colaboración al respecto y le agradecemos de antemano por su valiosa ayuda.

Nombre y apellidos: _____

Centro de trabajo: _____

Labor que realiza: _____Especialidad: _____ Años de experiencia: _____

Categoría docente: _____Categoría científica: _____

Publicaciones_____

Otros datos de interés: _____

Marque con una cruz según considere.

1. ¿Considera usted que el método propuesto para realizar la evaluación de la arquitectura se ajusta a las necesidades de los proyectos de Realidad Virtual?

   Si____ No____

2. ¿Cree usted que con la aplicación del método propuesto se logre incrementar la calidad del producto final?

   Si____ No____

3. ¿Considera usted que la evaluación de los atributos de calidad y las decisiones arquitectónicas son suficientes para el logro de los objetivos que propone el método?

   Si____ No____

Nota: En caso necesario especifique que elementos evaluaría.

_____

_____

_____

4. ¿Considera usted que son suficientes los pasos propuestos para evaluar la arquitectura.

   Si____ No____

¿Agregaría o eliminaría alguno? En caso necesario especifique que elemento agregaría o eliminaría.

_____

_____

_____

5. ¿Qué evaluación le otorgaría al método propuesto?

   Bien____ Regular____ Mal____

Le agradecemos cualquier sugerencia o recomendación sobre los aspectos señalados anteriormente.

Anexo 7: Respuesta de los especialistas

Para registrar los resultados de la validación de la propuesta se colocará un cero en las celdas donde la respuesta del especialista sea negativa y un uno en caso contrario.

Especialistas\Preguntas	Pregunta1	Pregunta2	Pregunta3	Pregunta4	Pregunta5
Especialista 1	1	1	1	1	Bien
Especialista 2	1	1	1	1	Bien
Especialista 3	1	1	1	1	Bien
Especialista 4	1	1	1	1	Bien
Especialista 5	1	1	1	1	Bien
Especialista 6	1	1	1	1	Bien
Especialista 7	1	1	1	1	Bien

GLOSARIO DE TÉRMINOS

ABD: Architecture Based Design.

ADD: Attribute-Driven Design Method.

ARD: Active Design Review.

ARID: Active Reviews for Intermediate Designs.

AS: Arquitectura de Software.

ASBC: Arquitectura de Software Basadas en Componentes.

ATAM: Architecture Tradeoffs Analysis Method.

CBAM: Cost-Benefit Analysis Method.

ERS: Especificaciones de Requisitos del Software.

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos.

ISO: Organización Internacional de Normalización.

ISO/IEC: Estándar internacional para la evaluación de la calidad de productos de software.

MECABIC: Método de Evaluación para Arquitecturas Basadas en Componentes.

MECASRV: Método para la evaluación de la calidad de la arquitectura en cuanto a Requerimientos No Funcionales en los software de Realidad Virtual.

QAW: Quality Attribute Workshops.

RF: Requisitos Funcionales.

RNF: Requisitos No Funcionales.

RV: Realidad Virtual.

SAAM: Software Architecture Analysis Method.

UCI: Universidad de las Ciencias Informáticas.