UNIVERSIDAD CENTROCCIDENTAL
"LISANDRO ALVARADO"

ESTUDIO DE ESTILOS EN LA ARQUITECTURA DEL SOFTWARE ORIENTADOS A HERRAMIENTAS E-LEARNING UTILIZANDO EL METODO SACAM

MARIA ELENA TORRES SAMUEL

Barquisimeto, 2006

UNIVERSIDAD CENTROCCIDENTAL “LISANDRO ALVARADO”
DECANATO DE CIENCIAS Y TECNOLOGÍA
MAESTRIA EN CIENCIAS DE LA COMPUTACION

ESTUDIO DE ESTILOS EN LA ARQUITECTURA DEL SOFTWARE ORIENTADOS A HERRAMIENTAS E-LEARNING UTILIZANDO EL METODO SACAM

Trabajo presentado para optar al grado de
Magíster Scientiarum en Ciencias de la Computación

Por: MARIA ELENA TORRES SAMUEL

Barquisimeto, 2006

ESTUDIO DE ESTILOS EN LA ARQUITECTURA DEL SOFTWARE ORIENTADOS A HERRAMIENTAS E-LEARNING UTILIZANDO EL METODO SACAM

Por: MARIA ELENA TORRES SAMUEL

Trabajo de Grado Aprobado

_____	_____
(Jurado 1)	**(Jurado 2)**
Tutor	

_____

(Jurado 3)

Barquisimeto, _____de _____de 2006

A Hermes Eduardo y a su futuro hermanito..

AGRADECIMIENTOS

A Dios.

A mis padres por estar conmigo siempre.

A mis hermanos por alentarme a seguir adelante, especialmente a Maritza por sus valiosos consejos.

A Henry por su apoyo incondicional.

A Lorena Del Favero, tutora de este trabajo de investigación, por su disposición y sus orientaciones.

Al profesor Luis Álvarez por el apoyo dado en todo momento y al profesor Edgar González por sus sugerencias y correcciones.

A los profesores de los seis decanatos de la UCLA.

Gracias.

INDICE GENERAL

Página

INDICE DE CUADROS

Página

INDICE DE FIGURAS

Página

INDICE DE GRAFICOS

Página

INDICE DE TABLAS

Página

UNIVERSIDAD CENTROOCIDENTAL “LISANDRO ALVARADO”
DECANATO DE CIENCIAS Y TECNOLOGIA
POSTGRADO EN CIENCIAS DE LA COMPUTACION

ESTUDIO DE ESTILOS EN LA ARQUITECTURA DEL SOFTWARE ORIENTADOS A HERRAMIENTAS E-LEARNING UTILIZANDO EL METODO SACAM

Autor (a): María Elena Torres Samuel
Tutor (a): Lorena Del Favero Cascone

RESUMEN

Un sistema puede estar conformado por una o más formas estructurales específicas conocidas como estilos arquitecturales de software. La presente investigación tiene como finalidad realizar un estudio de las Arquitecturas del Software desde el enfoque del modelo estructural, el cual se hace presente mediante los estilos arquitecturales, seleccionando el estilo más adecuado para herramientas e-learning, con base en los atributos de calidad deseables para estos sistemas. Se determinaron los elementos de algunos estilos arquitecturales y se seleccionaron dos arquitecturas como son: Arquitectura Orientada a Servicios y Arquitectura Basada en Componentes, los cuales aplican para sistemas de software distribuidos, posteriormente acorde al estándar ISO/IEC 9126(1991) se seleccionaron las características de calidad que aplican a Arquitecturas de Software. Finalmente se realizó una comparación entre las dos arquitecturas seleccionadas con los resultados de la evaluación obtenida en la fase diagnóstica mediante el método de comparación de arquitecturas (SACAM).

Palabras claves: e-learning, estilos arquitecturales, ISO/IEC 9126(1991), SACAM

INTRODUCCION

La importancia que actualmente tiene la administración de la información en las empresas e instituciones educativas ha generado un crecimiento del uso de herramientas tecnológicas que permitan distribuir y compartir dicha información. El uso de las herramientas e-learning como instrumento de expansión del conocimiento a través de la tecnología posibilita la formación continua bajo circunstancias en las que el tiempo, oportunidades y desplazamiento representan obstáculos que dificultan el desarrollo de un plan de formación tradicional continuado tanto en las organizaciones empresariales como educativas.

En el área de la educación, las herramientas e-learning hacen posible el avance y la continuidad en el desarrollo del paradigma de la educación a distancia. En este sentido, es necesario revisar cuales son las características de calidad que garanticen el funcionamiento adecuado de los componentes que la integran y cuál es el esquema ó Arquitectura de Software más adecuado para soportar las diversas funciones que una herramientas e-learning posee.

El área de la Arquitectura del Software proporciona a la Ingeniería del Software las bases para determinar si una Arquitectura de Software posee la capacidad suficiente para sostener la carga de trabajo, la flexibilidad de adaptarse a los cambios y a los requerimientos tecnológicos que el sistema requiere.

La presente investigación tiene como finalidad realizar un estudio de los estilos en la Arquitectura del Software orientados a herramientas e-learning y su posterior comparación mediante el método de comparación de Arquitecturas de Software SACAM. El estudio se encuentra dividido en capítulos, los cuales se detallan a continuación.

En el Capitulo I se plantea el problema observado que genera el trabajo investigativo, asimismo se describen los objetivos, la justificación e importancia, alcance y limitaciones representando estos aspectos las bases que conducen a la investigación.

En el Capitulo II se presentan los antecedentes encontrados en el área y la revisión teórica en los que se fundamenta la investigación, específicamente lo que respecta a e-learning, estilos en la Arquitectura de Software y estándares de calidad aplicados a Arquitecturas del Software así como también aspectos relacionados al método de comparación de arquitecturas SACAM.

En el Capitulo III se describen los aspectos metodológicos del presente estudio, se presenta además un análisis estadístico donde se indica la población y muestra objeto de estudio y las técnicas de recolección de datos para aplicar el instrumento de medición.

En el Capitulo IV se presentan los resultados obtenidos de la investigación a través de la revisión bibliografica y de la aplicación del instrumento de medición.

Finalmente en el Capitulo V se ofrece un conjunto de conclusiones y se plantean algunas recomendaciones enfocadas a las Arquitecturas de Software orientadas a herramientas e-learning.

CAPITULO I

EL PROBLEMA

Planteamiento del Problema

El aprendizaje de manera consciente o inconsciente siempre esta presente en la vida de toda persona, este proceso natural asistido mediante el uso de medios tecnológicos recibe un nombre: e-learning, su definición varía de acuerdo a las perspectivas de las políticas y estrategias determinadas, aunque poco a poco el significado y magnitud de dicho término ha adoptado diversos aspectos hasta convertirse en un elemento importante dentro de las empresas y en el ámbito profesional. Otero (2001)

Consecuentemente, para Silvio y Lapierre (2003), las herramientas e-learning han experimentado un gran crecimiento debido a la introducción de nuevas herramientas de informática y telecomunicaciones con cobertura global, y a la conciencia que han tomado los interesados de que ésta es una forma válida de educación, muy económica y de alta calidad que puede aumentar el valor del capital intelectual de las personas a gran escala, el elemento más importante de este crecimiento ha sido la rápida expansión del Internet, que combinada con el desarrollo de nuevas y más sofisticadas herramientas de aprendizaje a través de la red y facilidades de multimedia, ha producido un importante avance en las herramientas e-learning. En este sentido, Moneva (2003) expresa que los sistemas de gestión de la formación experimentarán una convergencia y se unirán áreas para formar entornos educativos más robustos.

Por su parte, según cifras de Staff High Tech Editores (2003), los ingresos mundiales del mercado de aprendizaje virtual son de aproximadamente 2,100 millones de dólares. El aprendizaje virtual comienza a ser un beneficio tangible en la infraestructura tecnológica y en los próximos años se volverá una forma de poner en uso programas de transferencia de conocimientos.

En efecto, el uso de Internet como medio en la formación ha provocado que numerosas organizaciones se hayan interesado en la implementación y difusión de sistemas de enseñanza. Internet se presenta como una respuesta lógica para cubrir las nuevas demandas de educación y formación. Según la consultora Nielsen/Netratings (2006), el constante crecimiento de Internet es cada vez mayor y en comparación con otros medios de comunicación como la televisión, la radio y la prensa escrita. El uso de Internet a nivel mundial y su crecimiento se presenta en la Tabla 1.

Tabla1: Estadística Mundial en el uso de Internet

Regiones	Poblacion (2006 Est.)	Poblacion Mundial	Usuarios Internet (Enero- 2006)	Crecimiento (2000-2005)
Africa	915,210,928	14.1 %	**22,737,500**	403.7 %
Asia	3,667,774,066	56.4 %	**364,270,713**	218.7 %
Europa	807,289,020	12.4 %	**290,121,957**	176.1 %
Oriente Medio	190,084,161	2.9 %	**18,203,500**	454.2 %
Norte America	331,473,276	5.1 %	**225,801,428**	108.9 %
Latinoamérica/Caribe	553,908,632	8.5 %	**79,033,597**	337.4 %
Oceania/ Australia	33,956,977	0.5 %	**17,690,762**	132.2 %
Total Mundial	6,499,697,060	100.0 %	**1,018,057,389**	182.0 %

Fuente: http://www.exitoexportador.com/stats.htm **Autor:** Nielsen/Netratings (2006)

De este modo, teniendo en cuenta la tendencia de crecimiento, las previsiones del mercado de las TIC en el presente año son positivas para Latinoamérica, lo cual proporciona un entorno favorable a las organizaciones para invertir en sistemas de enseñanza a distancia, ya que esta área sólo puede desarrollarse si se cuenta con el adecuado grado de desarrollo digital de la población. En relación a esto, el uso de e-

learning a nivel educativo en Latinoamérica como lo expresa Tapia Machain (2003) tiene presencia en las acciones educativas que se realizan en la región, específicamente en Venezuela su uso esta orientado hacia la capacitación universitaria y postgrado. Adicionalmente en el estudio realizado por Tapia Machain se describen las universidades que utilizan herramientas e-learning entre las cuales se encuentra la Universidad Centroccidental "Lisandro Alvarado" y su herramienta SABER. SABER (Sistema de Aprendizaje Basado en Redes), posee estudios que avalan su eficiencia en la administración de conocimiento y su facilidad de uso, entre los estudios realizados destaca el de Muñoz (2005).

Por lo tanto la existencia de factores de calidad es esencial en el uso de una herramienta de educación a distancia, proporcionando satisfacción para todas las personas que están interesadas en seguir una formación a través de esta modalidad. En consecuencia, es necesario revisar cuál es el esquema más adecuado para soportar las diversas funciones que una herramienta de educación a distancia debe poseer, en este sentido el esquema esta dado por la Arquitectura del Software.

La Arquitectura del Software es la organización fundamental de un sistema formado por sus componentes, las relaciones entre ellos, el contexto en el que se implementaran y los principios que orientan su diseño y evolución, IEEE (2000). En este orden de ideas, Dromey expresa, los proyectos de software exitosos están asociados con procesos sólidos y Arquitecturas de Software robustas. Dromey (1996)

Para Xiaofei y otros (2003), las herramientas e-learning han avanzado sin un esquema estandarizado de componentes donde se establezcan como están interrelacionados. La necesidad de tal arquitectura es crítica para definir áreas competitivas y definir estándares. Xiaofei y otros, afirman que existe una carencia de arquitecturas implementables para definir como combinar el modelo de información con el modelo de componentes y como definir una interface apropiada entre cada componente y subsistemas para alcanzar interoperabilidad.

De acuerdo a lo anterior, algunas propuestas han sido realizadas para promover una arquitectura basada en estándares de calidad que permita mejorar la interoperabilidad y la reusabilidad entre contenidos de aprendizaje y componentes de sistema, tal como lo indica el modelo WG01 de LSTC IEEE.

En tal sentido, se hace necesaria la oportuna selección de la arquitectura siguiendo métodos formales. El método SACAM proporciona un análisis razonado para un proceso de selección de la arquitectura comparando la aptitud de las arquitecturas candidatas a sistemas previstos.

Stoermer y otros (2003) expresan que; el método SACAM compara las arquitecturas basadas en un sistema de criterios derivados de las metas de negocio de una organización o sistema. SACAM fue desarrollado en un contexto técnico de la reutilización en donde una organización investigó concordancias arquitectónicas y las diferencias para explorar los diseños arquitectónicos.

Por lo antes expuesto, es factible realizar una comparación de arquitecturas que se orienten hacia herramientas e-learning basada en características de calidad utilizando el método SACAM. Adicionalmente, el área de Arquitectura del Software es un área reciente de la Ingeniería del Software y su aplicación en el desarrollo de sistemas es trascendental, es por ello que a continuación se plantean las siguientes interrogantes:

¿Cuales son los componentes que integran una herramienta e-learning?

¿Se considera importante estudiar cuales son los estilos arquitecturales en la Arquitectura del Software orientados hacia e-learning?

¿Cuales son los atributos de calidad en la Arquitectura del Software que debe poseer una herramienta e-learning?

¿Es relevante realizar un estudio comparativo entre algunas arquitecturas orientadas hacia e-learning basado en el método de comparación de arquitecturas de software SACAM?

Con el objeto de responder estas interrogantes, este trabajo propone realizar un estudio de los estilos arquitecturales en las Arquitecturas de Software orientados a herramientas e-learning utilizando el método de comparación de Arquitecturas de Software SACAM.

Objetivos

Objetivo General

Realizar un estudio de estilos en la Arquitectura del Software orientados a herramientas e-learning utilizando el método SACAM.

Objetivos Específicos

1. Identificar los componentes que integran una herramienta e-learning.
2. Estudiar los estilos en la Arquitectura del Software orientados hacia herramientas e-learning.
3. Determinar las características de calidad que debe poseer una Arquitectura de Software orientada a e-learning.
4. Realizar un análisis comparativo entre las diversas arquitecturas de software utilizando el método SACAM.

Justificación e Importancia

La incorporación del área de la Arquitectura del Software al diseño de sistemas es fundamental en la reducción de costos y fallas. La selección de una arquitectura adecuada o combinación de diferentes estilos arquitecturales es un aspecto clave para desarrollos de sistemas de software. Bass (1998)

Desde el enfoque institucional es importante la realización de este estudio debido a la contribución de nuevos conocimientos relativos a la Arquitectura del Software apoyando así la visión de "propiciar la generación, incorporación y difusión de nuevos conocimientos" definida por la Universidad Centroccidental "Lisandro Alvarado".UCLA (2005). Además de la contribución al área de investigación y soporte a la Universidad Virtual y su uso para investigaciones similares.

En consecuencia, se considera relevante la realización de este estudio ya que permitirá la ampliación de conocimientos en un área estratégica de la educación como es e-learning.

Alcance y Limitaciones

El objetivo de esta investigación tiene como finalidad realizar un estudio de los estilos en la Arquitectura del Software orientados a herramientas e-learning mediante el método de comparación de arquitecturas SACAM. El estudio se centra, en los estilos arquitecturales para sistema de software distribuidos como lo son la Arquitectura Orientada a Servicios y Arquitectura Basada en Componente, los cuales expresan esquemas de organización para sistemas de este tipo.

En cuanto a la selección del modelo de calidad para evaluar Arquitectura de Software esta se realizó, en función de las características a medir en tiempo de ejecución expresada por Preiss y otros (2000) y Bertoa y otros (2002). Adicionalmente se tomó como referencia a Losavio y otros (2001) en la selección de un modelo para evaluar calidad en Arquitecturas de Software, seleccionando el modelo el calidad ISO 9126 (1991) el cual tiene una alta relevancia en estudios de Arquitecturas de Software en comparación con modelos como Dromey y ABAS.

Finalmente, con el objetivo de determinar las características de calidad que una Arquitectura de Software orientada a herramienta e-learning debe poseer, la investigación abordará la opinión de los usuarios Profesores de la herramienta e-learning de la Universidad Centroccidental “Lisandro Alvarado” SABER.

CAPITULO II

MARCO TEORICO

Antecedentes

En el área de los estilos arquitecturales, uno de los acontecimientos arquitectónicos más importantes en el año 2000 fue la tesis de Fielding (2000), en la cual se presentó el modelo REST, y es allí donde se establece definitivamente el tema de las tecnologías de Internet y los modelos orientados a servicios y recursos en el centro de las preocupaciones de la disciplina. En el mismo año se publica la recomendación IEEE Std1471[1] que procura estandarizar la nomenclatura de descripción arquitectónica y homologa los estilos como un modelo fundamental de la representación conceptual.

Por otra parte, en el contexto de la Arquitectura de Software aplicada a herramientas e-learning, existen diversos estudios sobre el tema. En este orden de ideas, destaca la investigación de Anido y otros (2002) la cual propone una arquitectura de estándares abiertos para sistemas e-learning enfocándose en definir una arquitectura que promueva la interoperabilidad y la reusabilidad para sistemas e-learning distribuidos.

Dentro de este orden de ideas, WesterKamp (2003), expresa que los servicios de un entorno web proveen las funcionalidades necesarias para un sistema e-learning en lo que se refiere a atributos de reusabilidad e interoperabilidad lo cual permite la

[1] La recomendación IEEE Std 1471-2000 procura establecer una base común para la descripción de Arquitecturas de Software.

integración de aplicaciones; además de proveer la funcionalidad para implementar dispositivos móviles si existe un cliente apropiado para dicho dispositivo. Por su parte, Xiaohong (2003) propone el uso de SIMD (Single Intruction Multiple Data) y MIMD (Multiple Instruction Multiple Data) para generar una arquitectura de colaboración basada en un modelo de servicios Web para aplicaciones e-learning. En tanto, Sharma y Kitchens (2004), realizaron un análisis en torno a la Arquitectura Orientada a Servicios para el uso de dispositivos móviles en el aprendizaje a distancia. En relación a la aplicación del método SACAM, Schmitt (2005) realizó un estudio comparativo de Arquitecturas de Software orientadas a dispositivos móviles en base a los atributos de calidad Accesibilidad y Portabilidad.

Finalmente, en el ámbito nacional Angulo y otros (2002), realizaron un estudio de la Arquitectura de Software bajo un enfoque de calidad para un Sistema de Gestión de Conocimiento, el resultado obtenido de este estudio fue la selección de una arquitectura idónea para el dominio planteado basada en los atributos de calidad: Mantenibilidad, Confiabilidad y Eficiencia.

Es así como la bibliografía anteriormente citada corrobora la búsqueda de una arquitectura adecuada que integre calidad para la difusión de conocimientos, a través de la tecnología.

Bases teóricas

e-learning

El uso de e-learning ha revolucionado la forma en que la educación es impartida, en el modelo educativo tradicional el centro de la información es el docente, mediante la llegada la tecnología el centro de la información esta dado por la plataforma que ofrece los recursos educativos donde el profesor tiene a su disposición recursos para complementar sus clases; en ellos se provee al usuario de la herramienta de información relativa a: autoevaluaciones, foros de discusión, contenidos virtuales, enlaces a paginas Internet, bibliografía en línea, trabajos, proyectos y ejercicios.

En sentido amplio Rosenberg (2001); define e-learning como el uso de las tecnologías basadas en Internet para proporcionar un amplio despliegue de soluciones a fin de mejorar la adquisición de conocimientos y habilidades. El mismo autor establece tres criterios que se han de cumplir para poder aplicar correctamente el término:

1 Debe producirse en red, lo que permite una actualización inmediata, almacenamiento y recuperación, distribución y capacidad de compartir los contenidos y la información.
2 Debe llegar al usuario final a través de un computador, utilizando estándares tecnológicos de Internet.
3 Debe estar centrado en la visión más amplia de soluciones para el aprendizaje que van más allá de los paradigmas tradicionales de la formación.

De este modo Foix y Zavando (2002) resumen esquemáticamente los distintos componentes de una solución e-learning, la cual se puede observar en la Figura 1, estos son:

- Sistema de Administración del e-learning o LMS,
- Contenido
- Sistema de comunicación.

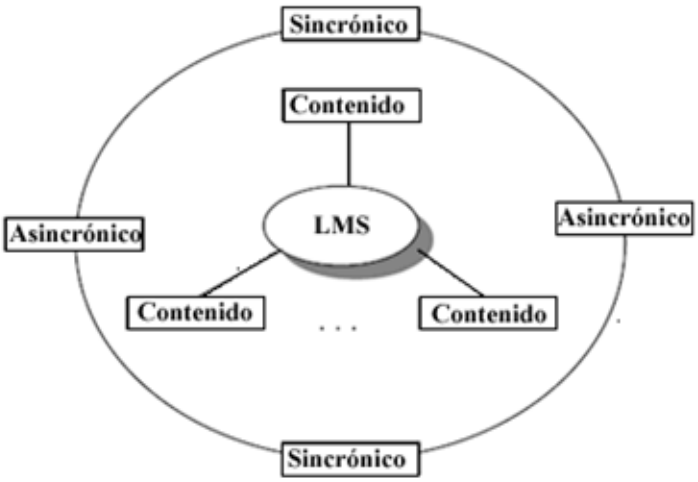

Figura 1: Elementos del e-learning
Fuente: Centro de Tecnología de Información Intec
Autor: Foix y Zavando (2002)

Sistema de Administración del e-learning o LMS

Es el núcleo alrededor del cual giran los demás elementos. Básicamente se trata de un software para servidores de Internet/Intranet que se ocupa de:

1 Gestionar los usuarios: inscripción, control de sus aprendizajes e historial, generación de informes, etc.
2 Gestionar y mantener los cursos, realizando un registro de la actividad del usuario: tanto los resultados de los tests y evaluaciones que realice, como de los tiempos y accesos al material formativo.

3 Gestionar los servicios de comunicación que son el apoyo al material en línea (online), foros de discusión, charlas, videoconferencia; programarlos y ofrecerlos conforme sean necesarios.

Contenido

Los contenidos para e-learning pueden estar en diversos formatos, en función de su adecuación a la materia tratada.

El más habitual es el WBT (Web Based Training), cursos online. Sin embargo, en otros casos puede tratarse de una sesión de "aula virtual", basada en videoconferencia y apoyada con una presentación en forma de diapositivas, en explicaciones en una "pizarra virtual". En este tipo de sesiones los usuarios interactúan con el docente, dado que son actividades sincrónicas en tiempo real. Otras veces el contenido no se presta a su presentación multimedia, por lo que se opta por materiales en forma de documentos que pueden ser descargados, complementados con actividades online tales como foros de discusión o charlas con los tutores.

Sistemas de comunicación sincrónica y asincrónica

El proceso de enseñanza-aprendizaje que permite que el e-learning pueda llevarse a cabo esta clasificada en dos formas: síncrona o en tiempo real y asíncrona o en tiempo diferido.

Síncrono

El e-learning síncrono es una modalidad de aprendizaje en la que el profesor y el alumno se escuchan, se leen y/o se ven en el mismo momento, independiente de que se encuentren en espacios físicos diferentes. Esto permite que la interacción se realice en tiempo real, como en una clase presencial. Para el soporte tecnológico de esta modalidad se cuenta con diferentes medios, entre los cuales se encuentran: Chat, aplicaciones compartidas y audio/videoconferencias entre otros.

Asíncrono

El e-learning asíncrono es una modalidad de aprendizaje en la que el profesor y el alumno interactúan en lugares diferentes y en tiempos distintos. Esto permite al alumno, a través de documentación, material y actividades en línea, entregados por el profesor, realizar su propio proceso de aprendizaje, planificando su ritmo y su tiempo de dedicación al estudio y de participación en tareas o actividades individuales o en grupo, sin necesidad de estar en conexión directa con los profesores y los otros alumnos. Las herramientas de comunicación o interacción más utilizadas para el apoyo de esta modalidad de aprendizaje son: e-mail, foros y listas de correo.

Componentes y sus estándares para herramientas e-learning

Según Xiaofei y otros (2003), los estándares establecidos para herramientas e learning se encuentran divididos en cinco categorías:

1 Metadatos: El contenido del e-learning debe ser etiquetado de una forma consistente que soporte la indexación, el almacenaje, búsqueda para múltiples herramientas para múltiples repositorios. Los datos usados para este propósito son referenciados como objetos metadata Existen algunas iniciativas para

crear estándares para metadatos tal como The Learning Object Metadata o LOM.de IEEE Learning Technology Standard y el Dublín Core Metadata.

2 Empaquetamiento de contenidos: las especificaciones de empaquetamiento de contenidos permiten transportar cursos de un sistema e-learning a otro. Las iniciativas en cuanto a la estandarización de contenidos incluyen: El IMS Content Packacking, El IMS Simple Sequencing Specification y el ADL Sharable Content Object Referente Model (SCORM).

3 Perfil de aprendizaje: El perfil de aprendizaje puede incluir datos personales, planes de aprendizaje, historial de aprendizaje, requerimientos de accesibilidad, certificaciones y grados. El mayor esfuerzo para estandarizar el perfil de aprendizaje proviene del IMS Learner Information Package (LIP).

4 Registro de aprendizaje: El registro de aprendizaje permite entregar y administrar componentes para conocer como estos deberían ser ofrecidos a los participantes del e-learning. Existen dos iniciativas actualmente que tratan con esos requerimientos: El IMS Enterprise Specification y Shools Interoperability Framework.

5 Comunicación entre contenido: Cuando el contenido es entregado, existe la necesidad de comunicar los datos aprendidos y la información de las actividades previas al contenido. Existe para el ello el ADL´s Sharable Content Object Referente Model (SCORM), proyecto basado sobre las especificaciones CMI de la Aviation Industry CBT Comitte

Ingeniería del Software

Booch (1998), Pressman(2002) y Juristo(2002), definen la ingeniería del software como una disciplina que abarca el desarrollo de sistemas complejos y de calidad, que requieren para su construcción de procesos rigurosos mediante modelos y estándares; adicionalmente describen los tres elementos principales en la Ingeniería de Software:

1 Métodos que indican como se debe construir el software
2 Herramientas que dan soporte al desarrollo del software
3 Procesos y metodologías, que corresponden a la unión entre los métodos, las entregas que se requieren, los controles necesarios para asegurar la calidad y la coordinación de los cambios.

Arquitectura del Software

Según Clements (1996):

> "La Arquitectura del Software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se le percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema." (p.56)

En una definición más amplia Garlan (2000), establece que las Arquitecturas del Software constituyen un puente entre el requerimiento y el código. Adicionalmente conforme al estándar IEEE 6.10.12.1990 la Arquitectura del Software se define como la aplicación de una estrategia sistemática, disciplinada y cuantificable al desarrollo, aplicación y mantenimiento del software, esto es, la aplicación de la Ingeniería del Software. En tal sentido, la Arquitectura de Software define los componentes (módulos, objetos, procesos, subsistemas, unidades de compilación, etc) y las asociaciones importantes entre ellos (invocaciones, envio de data, sincronización, dependencia, uso, entre otros).

En este orden de ideas, Shaw y Garlan (1996) proporcionaron un enfoque estructurado, explicando las diferencias entre definiciones en función de distintas clases de modelos. Destilando las definiciones y los puntos de vista implícitos o explícitos, los autores clasificaron estos modelos de la forma siguiente:

a) Modelos estructurales: Sostiene que la Arquitectura de Software está compuesta por componentes, conexiones entre ellos y otros aspectos tales como configuración, estilo, restricciones, semántica, análisis, propiedades, racionalizaciones, requerimientos, necesidades de los participantes. Esta área está caracterizada por el desarrollo de lenguajes de descripción arquitectónica (ADLs).
b) Modelos de framework**:** Son similares a la vista estructural, pero su énfasis primario radica en la estructura coherente del sistema completo, en vez de concentrarse en su composición. Los modelos de framework a menudo se refieren a dominios o clases de problemas específicos.
c) Modelos dinámicos: Enfatizan la cualidad conductual de los sistemas. Dinámico puede referirse a los cambios en la configuración del sistema, o a la dinámica involucrada en el progreso de la computación, tales como valores cambiantes de datos.
d) Modelos de proceso: Se concentran en la construcción de la arquitectura, y en los pasos o procesos involucrados en esa construcción. En esta perspectiva, la arquitectura es el resultado de seguir un argumento de proceso.
e) Modelos funcionales: Una minoría considera la arquitectura como un conjunto de componentes funcionales, organizados en capas que proporcionan servicios.

Los estilos arquitecturales se encuentran ubicados según la anterior clasificación dentro de los modelos estructurales, a tal fin, antes de comenzar con el estudio de los estilos arquitecturales se describirán los modelos de calidad aplicados a Arquitecturas del Software, comenzando por definir Calidad del Software.

Calidad del Software

Para definir Calidad del Software inicialmente ha de definirse el concepto de calidad, en relación a esto, existen diversas definiciones sobre calidad, algunas de las definiciones son las establecidas por los especialistas de la calidad, entre las cuales se tienen las siguientes:

- Adecuación para el uso a que se destina. Juran (1990).
- Contribución a la satisfacción de las necesidades. Deming (1989)
- Conjunto de propiedades y características de un producto o servicio que le confiere su capacidad para satisfacer necesidades expresadas o implícitas. ISO 9000 (2000).

Del mismo modo, en relación a la Calidad del Software, según IEEE, Std. 610-1990 (1990), se tiene la siguiente definición: "La calidad del software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario".

Por su parte, Pressman (2002) define la calidad como: "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" (p.135)

Estas definiciones coinciden en el hecho de que la calidad la define el receptor del objeto, servicio o software. Si lo que recibe responde a sus necesidades, expectativas o exigencias entonces existe calidad y el conjunto de características por las que le satisface representan el contenido de esa definición de calidad.

Estándares de calidad asociados a Arquitecturas de Software

La complejidad de los sistemas de software ha incrementado la necesidad de enfoques más rigurosos para el manejo de las decisiones en las etapas iniciales del diseño de software. En tal sentido, Dromey (1996) indica, la selección de una arquitectura o la combinación de diferentes estilos arquitecturales para un sistema de software es esencial para obtener sistemas de software robustos.

Es por esto que, la primera selección de una arquitectura puede realizarse definiendo prioridades sobre un escenario, las características de calidad relacionadas con el estilo y los patrones que constituyen el estilo, Losavio y otros (2004). En este sentido, el refinamiento se realiza basándose en los valores objetivos de los atributos de calidad asignados para algunas de las características de calidad, en relación a esto, Bass y otros (1998) expresan que las arquitecturas responden a requisitos de calidad específicos. De este modo, la arquitectura permite conocer cual es la calidad de un sistema y de sus atributos.

En consecuencia, Losavio y otros (2001) realizaron un estudio en base a tres modelos de evaluación de calidad para evaluar arquitecturas de software: ISO 9126(1991), Dromey y ABAS (Attribute-Based Architecturel Styles), su selección se basó en ellos ya que estos modelos comparten el hecho de considerar modelos de calidad relacionados con los producto que se obtienen en las primeras etapas del desarrollo, por lo tanto en base a esta investigación se describen a continuación estos tres modelos.

ISO/IEC 9126 (1991)

ISO/IEC 912(1991) define seis características de calidad y describe un modelo del proceso de evaluación del producto de software. Estas características se definen a continuación.

- Functionality (Funcionalidad): Es la capacidad del producto del software para proveer funciones que cumplan con necesidades específicas o implícitas, cuando el software es utilizado bajo ciertas condiciones
- Reliability (Confiabilidad): La confiabilidad es la capacidad del producto de software para mantener un nivel específico de rendimiento cuando es utilizado bajo condiciones específicas.
- Usability (Usabilidad): Esta categoría se refiere a la capacidad del producto de software para ser atractivo, entendido, aprendido y utilizado por el usuario bajo condiciones específicas.
- Efficiency (Eficiencia): La capacidad del producto de software para proveer un desempeño apropiado, relativo a la cantidad de recursos usados, bajo condiciones de estado.
- Mantenability (Mantenibilidad): La capacidad del producto de software para ser modificado, las modificaciones pueden incluir correcciones, mejoras, adaptaciones del software a cambios en el ambiente y especificaciones en los requerimientos funcionales.
- Portability (Portabilidad): La capacidad del producto de software para se transferido de un ambiente a otro, el ambiente puede incluir ambientes organizacionales, hardware o software.

Cada característica está relacionada con las sub-características que se van a evaluar. Ver Tabla 2.

Tabla 2: Características y sub-características del modelo de calidad ISO/IEC 9126 (1991)

Características de calidad	Sub-Característica
Funcionality (Funcionalidad)	Suitable (Idoneidad) Accuracy (Exactitud) Interoperabilty (Interoperatividad) Security (Seguridad) Compliance (Adherencia a normas)
Reliability (Confiabilidad)	Maturity (Madurez) Fault Tolerance (Tolerancia a Fallos) Recoverability (Capacidad de recuperación) Compliance (Adherencia a normas)
Usabilty (Usabilidad)	Understandability (Comprensibilidad) Learnability (Facilidad de aprendizaje) Operabilty (Operable) Compliance (Adherencia a normas)
Efficiency (Eficiencia)	Time Behavior (Comportamiento frente al tiempo) Resource Behavior (Uso de recursos) Compliance (Adherencia a normas)
Maintenability (Mantenibilidad)	Analyzability (Facilidad de análisis) Changeability (Capacidad de cambios) Stability (Estabilidad) Testeabilty (Facilidad para pruebas) Compliance (Adherencia a normas)
Portabilty (Portabilidad).	Adaptabilty (Adaptabilidad) Installabilty (Facilidad de instalación) Co-Existence (Co-existencia) Replaceabilty (Facilidad de reemplazo) Compliance (Adherencia a normas)

Fuente: Losavio y otros (2004)

Estas características son aplicables a la Arquitectura de Software ya que permiten indicar cual es la calidad de un sistema y de sus atributos, dentro de los sistemas complejos los atributos de calidad no pueden ser obtenidos por si solos, ya que la obtención de uno de ellos tendrá un efecto para la obtención de otros, es el caso de los atributos seguridad y confiabilidad los cuales se encuentran estrechamente relacionados,

En relación a la arquitectura, Losavio y otros (2004) expresan: “Funcionalidad, Confiabilidad, Eficiencia y Mantenibilidad tienen impacto directo sobre la

arquitectura "(p.30). Por otro lado, las sub-características de Mantenibilidad (Facilidad de análisis, Estabilidad y Facilidad para pruebas) dependen más de la metodología de desarrollo y del lenguaje de programación usado. Por ultimo, Portabilidad depende básicamente de las restricciones tecnológicas y Usabilidad esta relacionada con componentes de interfaces.

Por otro lado, las características de calidad pueden ser útiles no sólo para evaluar un producto de software, sino también para definir requisitos de calidad es por ello que posteriormente el ISO/IEC 9126 (1991) fue reemplazado por dos estándares:

1 ISO/IEC 9126 (2001): Software Engineering – Software Product Quality
2 ISO/IEC 14598 (1999): Information Technology – Software Product Evaluation.

Finalmente, Losavio y otros (2001) sugieren el uso del estándar ISO/IEC 14598 ya que este ofrece guías para la construcción o aplicación del modelo de calidad a diferencia del uso independiente del modelo ISO/IEC 9126 (2001) el cual no las proporciona.

Modelo de Calidad Dromey

Dromey define un modelo de calidad, donde hace énfasis en la definición apropiada de las características internas de un producto de software, sostiene que los componentes de software deben exhibir un conjunto completo y consistente de propiedades inherentes que resulten en manifestaciones de atributos de calidad, los pasos a seguir para la aplicación de este modelo son:

1 Identificar un conjunto de atributos de calidad de alto nivel para el producto.

2 Identificar los componentes del producto

3 Identificar y clasificar las propiedades de calidad inherentes tangibles más significativas de cada componente.

4 Proponer un conjunto de axiomas a través de los cuales se enlaza las propiedades con los atributos de calidad

5 Evaluar el modelo, identificar debilidades, refinarlo o desecharlo y empezar nuevamente.

ABAS

La arquitectura basada en atributos o ABAS fue propuesta por Klein y Klazman (1999) La intención de estos autores es asociar a la definición del estilo arquitectónico a un framework de razonamiento, ya sea cuantitativo o cualitativo, basado en modelos de atributos específicos.

El modelo de Klein y Kazman en realidad no tipifica como un estilo en estado puro, sino como una asociación entre la idea de estilo con análisis arquitectónico y atributos de calidad. En este contexto, los estilos arquitectónicos definen las condiciones en que han de ser usados. Además de especificar los habituales componentes y conectores, los estilos basados en atributos incluyen atributos de calidad específicos que declaran el comportamiento de los componentes en interacción.

Estilos Arquitecturales

En la publicación de Perry y Wolf (1992) se establece el razonamiento sobre estilos de arquitectura como uno de los aspectos fundamentales de la disciplina de la Arquitectura del Software. Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las

formas básicas posibles de estructuras de software, mientras que las formas complejas se articulan mediante composición de los estilos fundamentales.

Dentro de este marco, Shaw y Garlan (1996) definen un estilo arquitectural como una familia de sistemas en términos de patrones de organización estructural, un vocabulario de componentes y conectores con restricciones. Por su parte, Bass y otros (1998), determinaron que un estilo arquitectural está compuesto por un conjunto de componentes, una topología de cómo esos componentes se interrelacionan, restricciones semánticas y un conjunto de conectores. Siguiendo la misma línea, los estilos que habrán de describirse a continuación son los más representativos y vigentes dado que la agrupación de estilos y sub-estilos difieren de autor a autor, es por ello que se combinaron los principales estilos arquitecturales según Bass y otros (1998) y Fielding (2000), los cuales se muestran en la Tabla 3.

Tabla 3: Principales estilos arquitecturales

Estilo	Composición
Estilos de Flujo de Datos	Tubería y filtros
Estilo de Llamada y Retorno	Model-View-Controller (MVC) Arquitectura en Capas Arquitectura Orientada a Objetos Arquitectura Basada en Componentes
Estilos Centrados en Datos	Arquitectura de Pizarra o Repositorio
Estilos de Código Móvil	Arquitectura de Máquinas Virtuales
Estilos heterogéneos	Arquitectura Basada en Atributos
Estilos Peer-to-Peer	Arquitectura Basada en Eventos Arquitectura Orientada a Servicios Arquitectura Basada en Recursos

Fuente: La autora de la investigación con recopilación de la información obtenida durante la investigación.

A continuación se describe cada uno de los estilos arquitecturales descritos en la Tabla 3.

Estilos de Flujos de Datos

Esta familia de estilos enfatiza la reutilización y la posibilidad de cambios para sistemas que implementan transformaciones de datos en pasos sucesivos.

Tubería y filtros

Históricamente, este tipo de arquitectura se relaciona con las redes de proceso descritas por Kahn hacia 1974 y con el Proceso Secuencial Comunicantes (CSP) ideados cuatro años más tarde. Ha prevalecido el nombre de tubería-filtros ya que ejecutan formas variables de transformación, una de las cuales puede ser el filtrado. En uno de los trabajos recientes más completos sobre este estilo, Doberkat (2002) lo define en estos términos:

> "Una tubería (pipeline) es una arquitectura que conecta componentes computacionales (filtros) a través de conectores (pipes), de modo que los cálculos se ejecutan a la manera de un flujo. Los datos se transportan a través de las tuberías entre los filtros, transformando gradualmente las entradas en salidas." (p.1)

El sistema tubería-filtros se percibe como una serie de transformaciones sobre sucesivas piezas de los datos de entrada. Los datos entran al sistema y fluyen a través de los componentes.

Estilos de Llamada y Retorno

Esta familia de estilos enfatiza la capacidad de cambios y la escalabilidad. Son los estilos más generalizados en sistemas en gran escala. Miembros de la familia son las arquitecturas de programa principal y subrutina

Model View Controller (MVC)

El MVC es propio de las aplicaciones en Smalltalk por lo menos desde 1992, antes que se generalizaran las arquitecturas en capas múltiples. Según Burbeck (1992), el estilo Modelo-Vista-Controlador (MVC) separa el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres clases diferentes los cuales se describen a continuación:

1. Modelo. El modelo administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado y responde a instrucciones de cambiar el estado
2. Vista. Maneja la visualización de la información.
3. Controlador. Interpreta las acciones de los periféricos, informando al modelo y/o a la vista para que cambien según resulte apropiado.

Arquitectura en Capas

Garlan y Shaw (1994), definen el estilo en capas como una organización jerárquica tal que cada capa proporciona servicios a la capa inmediatamente superior y utiliza las funciones que le brinda la inmediatamente inferior. En un estilo en capas, los conectores se definen mediante los protocolos que determinan las formas de la

interacción. Las restricciones topológicas del estilo pueden incluir una limitación, más o menos rigurosa, que exige a cada capa operar sólo con capas adyacentes, y a los elementos de una capa entenderse sólo con otros elementos de la misma. De esta forma, este estilo soporta un diseño basado en niveles de abstracción crecientes, lo cual a su vez permite a los implementadores la partición de un problema complejo en una secuencia de pasos incrementales admite además optimizaciones y refinamientos, y proporciona amplia reutilización.

Arquitectura Orientada a Objetos (OO)

Los componentes de este estilo son los objetos, o más bien instancias de los tipos de dato abstractos. Un rasgo importante de este aspecto es que la representación interna de un objeto no es accesible desde otros objetos. Las características de las arquitecturas OO, se resumen de la siguiente forma:

- Los componentes del estilo se basan en principios OO: encapsulamiento, herencia y polimorfismo. Son asimismo las unidades de modelado, diseño e implementación, los objetos y sus interacciones es el centro del diseño de la arquitectura y de la estructura de la aplicación.
- Las interfaces están separadas de las implementaciones. En general la distribución de objetos es transparente, y en el estado de arte de la tecnología no es relevante si los objetos son locales o remotos. El mejor ejemplo de OO para sistemas distribuidos es Common Object Request Broker Architecture (CORBA), en la cual las interfaces se definen mediante Interface Description Language (IDL); un Object Request Broker media las interacciones entre objetos clientes y objetos servidores en ambientes distribuidos.
- En cuanto a las restricciones, puede admitirse o no que una interfaz pueda ser implementada por múltiples clases.

Como limitación, la principal desventaja de este estilo se manifiesta en el hecho de que para poder interactuar con otro objeto a través de una invocación de procedimiento, se debe conocer su identidad. Esta situación contrasta con lo que es el caso en estilos tubería-filtros, donde los filtros no necesitan poseer información sobre los otros filtros que constituyen el sistema. La consecuencia inmediata de esta característica es que cuando se modifica un objeto se deben modificar también todos los objetos que lo invocan. También se presentan problemas de efectos colaterales en cascada: si A usa B y C también lo usa, el efecto de C sobre B puede afectar a A.

En la literatura sobre estilos, las arquitecturas orientadas a objeto han sido clasificadas de formas diferentes conforme a los diferentes puntos de vista que alternativamente enfatizan la jerarquía de componentes, su distribución topológica o las variedades de conectores.

Arquitectura Basada en Componentes

Existen diversas definiciones de componentes, según Szyperski (1995), un componente de software es una unidad de composición con interfaces especificas contractualmente y con dependencias del contexto explícitas. En un estilo arquitectural de esta clase se tiene:

a) Los componentes son las unidades de modelado, diseño e implementación.
b) Las interfaces están separadas de las implementaciones, y las interfaces y sus interacciones son el centro del diseño arquitectónico. Los componentes soportan algún régimen de introspección, de modo que su funcionalidad y propiedades pueden ser descubiertas y utilizadas en tiempo de ejecución.
c) En cuanto a las restricciones, puede admitirse que una interfaz sea implementada por múltiples componentes. Usualmente, los estados de un componente no son accesibles desde el exterior.

Estilos Centrados en Datos

Esta familia de estilos enfatiza la integrabilidad de los datos. Se estima apropiada para sistemas que se fundamentan en el acceso y actualización de datos en estructuras de almacenamiento.

Arquitectura de Pizarra o Repositorio

En esta arquitectura hay dos componentes principales: una estructura de datos que representa el estado actual y una colección de componentes independientes que operan sobre él. Shaw y Garlan (1996).

Estos sistemas se han usado en aplicaciones que requieren complejas interpretaciones de proceso de señales o en sistemas que involucran acceso compartido a datos con agentes débilmente acoplados. También se han implementado estilos de este tipo en procesos en lotes de base de datos y ambientes de programación organizados como colecciones de herramientas en torno a un repositorio común. Un sistema de pizarra se implementa para resolver problemas en los cuales las entidades individuales se manifiestan incapaces de aproximarse a una solución, o para los que no existe una solución analítica.

Estilos de Código Móvil

Arquitectura de Máquina Virtual

Según Bass (1998), "una maquina virtual tiene como objetivo alcanzar la calidad en portabilidad, las cuales simulan algunas funcionalidad que no están nativas en el software o en el hardware donde se implementan" (p.98). La arquitectura de

máquinas virtuales se ha llamado también intérpretes basados en tablas. Es así como, todo intérprete involucra una máquina virtual implementada en software. De este modo, un intérprete posee por lo general cuatro componentes:

1 Una máquina de interpretación que lleva a cabo la tarea,
2 Una memoria que contiene el seudo-código a interpretar,
3 Una representación del estado de control de la máquina de interpretación, y
4 Una representación del estado actual del programa que se simula.

Estilos Heterogéneos

Arquitectura Basada en Atributos

La arquitectura Basada en Atributos o ABAS fue propuesta por Klein y Klazman (1999). La intención de estos autores es asociar a la definición del estilo arquitectónico un framework de razonamiento, ya sea cuantitativo o cualitativo, basado en modelos de atributos específicos. Su objetivo se fundamenta en la premisa que dicha asociación proporciona las bases para crear una disciplina de diseño arquitectónico y con ello se lograría que la Arquitectura de Software estuviese más cerca de ser una disciplina de ingeniería. Además de especificar los habituales componentes y conectores, los estilos basados en atributos incluyen atributos de calidad específicos que declaran el comportamiento de los componentes en interacción.

Estilos Peer-to-Peer

Esta familia, también llamada de componentes independientes, enfatiza la Capacidad de cambios por medio de la separación de las diversas partes que intervienen en la computación. Consiste por lo general en procesos independientes o entidades que se comunican a través de mensajes. Cada entidad puede enviar mensajes a otras entidades, pero no controlarlas directamente, entre estos estilos se encuentran las arquitecturas basadas en eventos, orientadas a servicios y orientadas a recursos.

Arquitectura Basada en Eventos

Las arquitecturas Basadas en Eventos se vinculan históricamente con sistemas basados en actores, y redes de conmutación de paquetes (publicación/suscripción). Los conectores de estos sistemas incluyen procedimientos de llamada tradicionales y vínculos entre anuncios de eventos e invocación de procedimientos. La idea dominante en la invocación implícita es que, en lugar de invocar un procedimiento en forma directa un componente puede anunciar mediante difusión uno o más eventos. Un componente de un sistema puede anunciar su interés en un evento determinado asociando un procedimiento con la manifestación de dicho evento.

Un estilo perteneciente a esta clase es C2 o Chiron-2. Una aplicación de arquitectura C2 está constituida por componentes que se comunican a través de buses; la comunicación está basada en eventos. Un componente puede enviar o recibir eventos hacia o desde los buses a los que está conectado. Componentes y buses se pueden componer topológicamente de distintas maneras, siguiendo reglas y restricciones particulares.

Arquitectura Orientada a Recursos

Fielding (2000) considera REST (Representational State Transfer) como resultado de la composición de varios estilos más básicos, incluyendo repositorio replicado, sistema en capas, máquina virtual, entre otros. Fielding no solamente expande más allá de lo habitual y quizá más de lo prudente el catálogo de estilos existentes, sino que su tratamiento estilístico se basa en Perry y Wolf (1992) antes que en Garlan y Shaw (1994), debido a que la literatura sobre estilos que se deriva de este último texto sólo considera elementos, conectores y restricciones, sin tomar en consideración los datos, que para el caso de REST al menos constituyen una dimensión esencial. En este sentido, REST constituye un ejemplo de referencia para derivar descripciones de arquitecturas de la vida real en base a un procedimiento de composición estilística.

Arquitectura Orientada a Servicios

Estos estilos arquitecturales actualmente se conocen como SOA y pertenece a un campo mayor que se denomina Web Service. Web Service es un sistema de software diseñado para soportar interacción máquina-a-máquina sobre una red.

En la literatura clásica referida a estilos, las arquitecturas basadas en servicios pueden clasificarse con lo que Garlan y Shaw (1994) definen como el estilo de procesos distribuidos. Según esta perspectiva existen dos variantes del estilo:

a) Participantes especificados (named): Estilo de proceso de comunicación. El ejemplar más conocido sería el modelo cliente-servidor. Si el servidor trabaja sincrónicamente, retorna control al cliente junto con los datos; si lo hace asincrónicamente, sólo retorna los datos al cliente, el cual mantiene su propio hilo de control.

b) Participantes no especificados (unnamed): Paradigma publish/subscribe, o estilo de eventos.

Cabe considerar por otra parte, lo que hace diferentes a la Arquitectura Orientada a Servicios de otros mecanismos como CORBA o DCOM es que utiliza estándares de Web para los formatos de datos y los protocolos de aplicación. Esto permite que las aplicaciones interoperen con mayor libertad.

La arquitectura funciona de la siguiente manera:

1 Un servicio, que es una entidad de software la cual encapsula funcionalidad de negocios y proporciona dicha funcionalidad a otras entidades a través de interfaces públicas bien definidas.
2 Los componentes del estilo (o sea los servicios), están débilmente acoplados. El servicio puede recibir requerimientos de cualquier origen. La funcionalidad del servicio se puede ampliar o modificar sin rendir cuentas a quienes lo requieran. Los servicios son las unidades de implementación, diseño e implementación. Los componentes que requieran un servicio pueden descubrirlo y utilizarlo dinámicamente mediante UDDI y sus estándares sucesores. En general, aunque hay alternativas, no se mantiene persistencia de estado y tampoco se pretende que un servicio recuerde nada entre un requerimiento y el siguiente.

En la figura 2 se muestran los principales elementos que conforman una Arquitectura Orientada a Servicios.

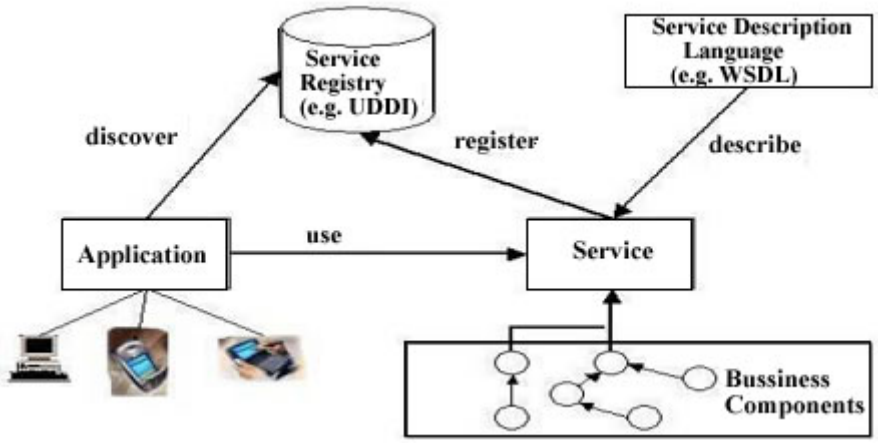

Figura 2: Elementos claves en una Arquitectura Orientada a Servicios
Fuente: Architecture paradigms and their influences and impacts on component-based software systems
Autor: Guijun Wang y Casey Fun (2004)

SACAM (Software Architecture Comparison Analysis Method)

Las Arquitecturas de Software se diseñan con requisitos y elementos particulares, sin embargo, las organizaciones necesitan a menudo seleccionar una Arquitectura del Software para su desarrollo futuro de varias arquitecturas candidatas. Stoermer (2003)

El método del análisis de la comparación de la Arquitectura del Software (SACAM) fue creado para proporcionar el análisis razonado para un proceso de selección de la arquitectura comparando la aptitud de los candidatos de la arquitectura a los requerimientos del sistema.

SACAM compara las arquitecturas basadas en un sistema de criterios derivados de las metas de negocio de una organización. El SACAM fue desarrollado en un contexto técnico de la reutilización en donde una organización investigó concordancias arquitectónicas y las diferencias para explorar los diseños

arquitectónicos para un producto de software en relación a su arquitectura. SACAM comprende los pasos siguientes:

a) La preparación: identifica las metas de negocio relevantes necesarias en la comparación y examina la documentación disponible para cada candidato de la arquitectura.
b) La selección de los criterios: deriva los criterios de la comparación de las metas de negocio y las refina a ellos en escenarios con atributos de calidad
c) La determinación de los directorios de la extracción determina las visiones, la táctica, los estilos, y los patrones arquitectónicos que se buscan durante las extracciones siguientes para encontrar la evidencia de soporte para los escenarios del paso b.
d) La extracción de la visión y de los indicadores extrae las visiones arquitectónicas para cada candidato según los directorios de la extracción del paso c.
e) Detecta los indicadores que apoyan los escenarios de los atributos de calidad del paso b.
f) Se resumen los resultados del análisis y se proporciona una recomendación para el procedimiento de toma de decisión.

Operacionalización de las Variables

Variable Conceptual

Calidad en Arquitectura de Software

Variables Operacionales

Para efectos de la operacionalización de la variable conceptual se procedió de la siguiente manera:

La variable Calidad en Arquitectura de Software se operacionalizó considerando los atributos de calidad deseables que una Arquitectura de Software orientada a e-learning debería satisfacer.

Las dimensiones a considerar comprenden las características de calidad cuyos indicadores son las sub-características de calidad. De acuerdo a las características del modelo de calidad ISO/IEC 9126(1991) se estudiaron dichas características en base a Losavio y otros (2004), la cual se presenta a continuación en la Tabla 4.

Tabla 4: Características de calidad para Arquitecturas de Software

Características de calidad	Sub-Características
Funcionality (Funcionalidad)	Suitable (Idoneidad) Accuracy (Exactitud) Interoperabilty (Interoperatividad) Security (Seguridad) Compliance(Adherencia a normas)
Reliability (Confiabilidad)	Maturity (Madurez) Fault Tolerance (Tolerancia a Fallos) Recoverability (Capacidad de recuperación)
Usabilty (Usabilidad)	Understandability (Comprensibilidad) Learnability (Facilidad de aprendizaje) Operabilty (Operable)
Efficiency (Eficiencia)	Time Behavior (Comportamiento frente al tiempo) Resource Behavior (Uso de recursos)
Maintenability (Mantenibilidad)	Analyzability (Facilidad de análisis) Changeability (Capacidad de cambios) Stability (Estabilidad) Testeabilty (Facilidad para pruebas)
Portabilty (Portabilidad)	Adaptabilty (Adaptabilidad) Installabilty (Facilidad de instalación) Co-Existence (Co-existencia) Replaceabilty (Facilidad de reemplazo)

Fuente: Losavio y otros (2004)

El proceso de selección de características y sub-características a evaluar se basó en Losavio y otros (2004), en el cual se realiza un estudio de las características de calidad asociadas a arquitecturas de software, las características y sub-características resaltadas en la Tabla 4 obedecen a los siguientes criterios:

- Las sub-características de mantenibilidad (Facilidad de análisis, Estabilidad y Facilidad para pruebas) dependen más de la metodología de desarrollo y del lenguaje de programación usado.
- Portabilidad depende básicamente de las restricciones tecnológicas.
- Usabilidad esta relacionada a componentes de interfaces.
- En tanto que Funcionalidad, Confiabilidad, Eficiencia y Mantenibilidad impactan directamente sobre la arquitectura.

Adicionalmente, Preiss y otros (2000) expresan, el instante en el cual una característica puede ser observada o medida, permite establecer otra clasificación. Así se tienen dos posibles categorías dependiendo de sí la característica es observable en tiempo de ejecución, por ejemplo rendimiento o durante el ciclo de vida del producto, ejemplo mantenibilidad. Según la clasificación de Bertoa y otros (2002) se puede apreciar en la Tabla 5 estas características, a tal fin se seleccionaron las características medibles en tiempo de ejecución, ya que estas son apreciables directamente por el usuario al usar el sistema de software y la fase diagnóstica esta dirigida a este tipo de usuarios.

Tabla 5: Clasificación de características de calidad para sistemas de software

Característica	Subcaracterística (tiempo ejecución)	Subcaracterística (ciclo de vida)
Funcionalidad	Precisión Seguridad	Idoneidad Interoperatividad Conformidad
Fiabilidad	Recuperabilidad	Madurez
Facilidad de Uso		**Facilidad de Aprendizaje** **Facilidad de Comprensión** **Operatividad**
Eficiencia	Comportamiento Temporal Utilización de Recursos	
Mantenibilidad		Cambiabilidad Facilidad de Prueba
Portabilidad		Reemplazabilidad

Fuente: Bertoa y otros (2002)

De esta manera, de acuerdo a las características de calidad seleccionadas se presenta a continuación un resumen de la operacionalización de la variable en estudio

CAPITULO III

MARCO METODOLOGICO

Naturaleza del Estudio

La presente investigación se orienta hacia una investigación de tipo monográfica documental, ya que se realizará un estudio descriptivo o diagnostico de una situación inherente a los estilos arquitecturales de software orientados a herramientas e-learning, tal estudio lleva a la descripción o evaluación de los elementos que configuran el ámbito del problema. Esta investigación se basa principalmente en fuentes bibliograficas, documentales y estudios comparados de análisis de problemas que ocurren en la práctica. (Manual para la presentación del Trabajo Conducente al Grado Académico de: Especialización – Maestría- Doctorado de la Universidad Centroccidental Lisandro Alvarado, 2002).

Fases del estudio

En atención a esta modalidad de investigación y con el fin de cumplir con los requisitos involucrados; el estudio comprende tres fases.

Fase 1:

Recopilación y revisión bibliográfica:

1. Recopilación de información relativa a los componentes que integran una

herramienta e-learning.

2. Revisión bibliografica de estilos arquitecturales asociados a herramientas e-learning, específicamente los asociados a las siguientes arquitecturas: Basada en Componentes y Orientada a Servicios, de este paso se obtendrá información detallada y especifica de las arquitecturas objeto de estudio.

Fase 2:

Fase Diagnóstica:

Con la finalidad de obtener información sobre las características de calidad en arquitectura en esta fase se procederá al diseño del instrumento para conocer la relevancia de estas características en una herramienta e-learning. A tal fin se seguirá la metodología que provee el estándar ISO/IEC 9126 (1991) conjuntamente con el estándar ISO/IEC 14598-5. Adicionalmente se realizaran las siguientes etapas:

- Diseño del instrumento de medición
- Determinar la validez del instrumento
- Analizar los resultados de la validación
- Aplicar el instrumento de medición
- Determinar la confiabilidad
- Analizar los datos recabados por el instrumento de medición
- Presentar conclusiones del diagnóstico

Fase 3:

Comparación.

En esta fase se realizará un análisis comparativo de las dos arquitecturas seleccionadas en la fase 1 utilizando los atributos de calidad obtenidos en la fase 2.

Población y muestra

La población objeto de estudio estuvo constituida por docentes usuarios de la herramienta e-learning SABER de la UCLA. Según información suministrada por el Profesor Alvaro Muñoz, docente de la UCLA, los usuarios de la herramienta SABER son: Administrador, Supervisor, Profesor y Estudiante. A tal fin, se seleccionó la población constituida por los Profesores.

La muestra según Hernández (1998) es esencia, un subgrupo de la población, es un subconjunto de elementos que pertenecen a ese conjunto definido en sus características denominado población. El tipo de selección de la muestra es probabilística, las cuales tienen como ventaja principal que puede medirse el tamaño del error en la predicción. Puede decirse incluso que el principal objetivo en el diseño de una muestra es reducir al mínimo este error al que se llama error estándar.

La población de Profesores usuarios de la herramienta SABER según la Analista de Sistemas Melina Rodríguez, Administradora de la herramienta e-learning SABER de la UCLA se encuentra constituida por 88 profesores. La muestra objeto de estudio fue de 26 profesores, el tamaño de la muestra se cálculo en base a las siguientes variables:

N = Tamaño de la población

$\bar{y}$ = Valor promedio de una variable = 1

Se = error estandar =0.05 seleccionado

V^2 = Varianza de la población. Su definición (Se) cuadrado del error estándar

S^s = Varianza de la muestra expresada como la probabilidad de ocurrencia de $\bar{y}$.

n'= tamaño de la muestra sin ajustar

n = tamaño de la muestra

Sustituyendo tenemos que:

$n' = \frac{S^2}{V^2}$, donde

$S^s = p(1-p) = 0.90(1-0.90) = 0.09$

$V = (0.05)^s = 0.0025$

$n' = \frac{0.09}{0.0025} = 36$

$$n = \frac{n'}{1+\frac{n}{N}} = \frac{36}{1+\frac{36}{88}} = \frac{36}{1.4090} = 25.55 \cong 26 \text{ profesores}$$

Es necesario además estratificar la muestra en relación a las categorías que se presentan en la población, se divide la población en subpoblaciones y se selecciona una muestra por cada estrato, la población se encuentra estratificada en seis (06) Decanatos, geográficamente ubicadas en distintas zonas de la ciudad de Barquisimeto. Por tal motivo, la muestra está conformada proporcionalmente por profesores de cada Decanato, esto se obtuvo calculando el factor de proporción (Fp) de la muestra con respecto a la población, esto es:

$$Fp = \frac{n}{N} = \frac{26}{88} = 0.2954$$

La información referente a los estratos por Decanato se muestra en la tabla siguiente:

Tabla 7: Tamaño de la muestra por Decanato

Decanato (Estrato)	**Nro de Profesores de la población (Ni)**	**Tamaño de la muestra del estrato, Fp=0.2954 ni = (Ni x Fp)**	**Porcentaje**
Administración y Contaduría	24	07	26,92 %
Agronomía	10	03	11,54 %
Ciencia y Tecnología	28	08	30,77 %
Ingeniería Civil	06	02	11,54 %
Medicina	11	03	11,54 %
Veterinaria	09	03	7, 69 %
TOTALES	N=88	n=26	100 %

Fuente: La autora de la investigación

Técnicas e Instrumentos de Recolección de Datos

El plan de recolección de datos se llevó a cabo en las siguientes etapas:

Primera Etapa: Diseño del Instrumento. Se diseñó un instrumento contentivo de 20 items distribuido en tres (3) partes o dimensiones con escalamiento de tipo Licker. A nivel general todas las dimensiones son valoradas según la escala de cinco (05) puntos de acuerdo a las categorías: Indispensable (5), Sumamente importante (4), Medianamente importante (3), Poco Importante (2) y No se toma en cuenta (1). Su objetivo fue el de diagnósticar los atributos de calidad que deben están presentes en una Arquitectura de Software orientada a herramientas e-learning. Para el logro de tal objetivo la primera parte consta de seis (06) ítems los cuales buscan obtener de los encuestados sus opiniones sobre atributos de funcionalidad. En la segunda parte de

dicho cuestionario se consideran seis (06) ítems los cuales buscan obtener de los encuestados opiniones sobre los atributos de confiabiabilidad. Finalmente la tercera parte contiene ocho (08) ítems que permiten evaluar los atributos relacionados a eficiencia en herramientas e-learning.

Segunda Etapa: Validez del Instrumento. La validez del instrumento se hizo a través de la evaluación por parte de los expertos quienes pudieron determinar la relación que existe entre los objetivos del estudio, las variables, dimensiones e indicadores que lo conforman y los ítems que contienen los instrumentos. A tal fin a cada experto se le entregó una carta de solicitud de validación (Ver Anexo 2), un ejemplar del cuestionario (Ver Anexo 1), la operacionalización de variables en estudio (Ver Tabla 4) y el formato para la evaluación de la pertinencia, claridad y congruencia de cada ítem en relación a las dimensiones y sub-características definidas (Anexo 3)

Los expertos estuvieron integrados por los Profesores Luís Alvarez y Edgar González docentes de la Universidad Centroccidental "Lisandro Alvarado", y la Profesora Nohelia Flores, docente de la Universidad "Simón Bolívar".

Tercera Etapa: Análisis de los resultados de la validación. En la tercera etapa se analizaron los resultados de la validación del instrumento. Se procedió a realizar las modificaciones que permitieron mejorar la calidad para obtener los objetivos percibidos por dicho instrumento. Entre las observaciones realizadas estuvieron reformulación de planteamientos y especificación del nivel de detalle para mejorar la claridad y congruencia de los planteamientos formulados.

Cuarta Etapa: Aplicación del instrumento. Se aplicó el instrumento a los docentes de la UCLA seleccionados de la población, a los cuales se contactó y se les aplicó el instrumento (Ver Anexo 1).

Quinta Etapa: Confiabilidad del Instrumento. Esta etapa consistió en determinar la confiabilidad de los resultados del cuestionario, se efectuó una prueba piloto de tamaño del 10 % de la población. En la presente investigación, se procedió a determinar la confiabilidad de consistencia interna del instrumento de Alpha de Cronbach calculado con el software SPSS 7.5 para Windows, obteniéndose el siguiente resultado:

Tabla 8: Resultados del análisis de confiabiabilidad usando el coeficiente Alfa de Cronbach

N	Nro de sujetos	Nro de Items	Valor obtenido
88	10% n=09	20	$\alpha = 0.8651$

Fuente: La autora de la investigación

El uso del coeficiente Alfa de Cronbach requiere una sola administración del instrumento de medición, su valor oscila entre 0 y 1, en la medida que aumenta su valor mayor es la consistencia interna de la escala y, en consecuencia, menor es la varianza. Cronbach (1951). Según la tabla 8, el valor obtenido para α es de 0.8651 por lo tanto el instrumento de medición es altamente confiable.

Las etapas de análisis de los datos recabados por el instrumento de medición y la presentación de las conclusiones del diagnóstico se presentan en el Capitulo IV como parte de los resultados del trabajo de investigación.

CAPITULO IV

RESULTADOS

Introducción

Atendiendo a la aplicación del método SACAM en el estudio de estilos arquitecturales en las Arquitecturas de Software, el presente trabajo de investigación comprende la revisión bibliografica de los componentes de una herramienta e-learning así como también los estilos arquitecturales aplicables a este tipo de sistemas. Esta revisión permite derivar los criterios de las metas de negocio y examina la documentación disponible de las Arquitecturas de Software candidatas, siendo este el paso inicial en la aplicación del método SACAM.

Posteriormente, mediante los resultados de la encuesta diagnóstica sobre calidad en Arquitectura de Software se determinan los atributos de calidad que son relevantes y aplicables a herramientas e-learning, tal como lo indican los pasos subsiguientes del método. Finalmente, se realiza una comparación de las Arquitecturas de Software seleccionadas, resumiendo los resultados y proporcionando una recomendación como se indica en el paso final del método SACAM.

Componentes que integran una herramienta e-learning

Un curso online involucra el uso de tipos especiales de web sites usualmente llamados web site instruccionales, el cual es un ambiente para enseñanza-aprendizaje implementado y usado mediante la tecnología web.

El ambiente de un curso de enseñanza-aprendizaje en línea puede ser comprendido desde cuatro diferentes perspectivas o vistas: instruccional, tecnológico, estructural y funcional. Montilva y Sandia (2004)

Vista Instruccional

Define el modo de interacción o comunicación entre los participantes y componentes de la herramienta e-learning. El modo de interacción puede ser sincrónico o asincrónico

Vista Tecnológica

Desde esta perspectiva el curso es visto como una selección de páginas web ínter enlazadas en un servidor web y accedidas desde cualquier computadora cliente conectada a Internet. En tal sentido una arquitectura general para herramientas e-learning consta de tres capas, como se muestra en la figura siguiente:

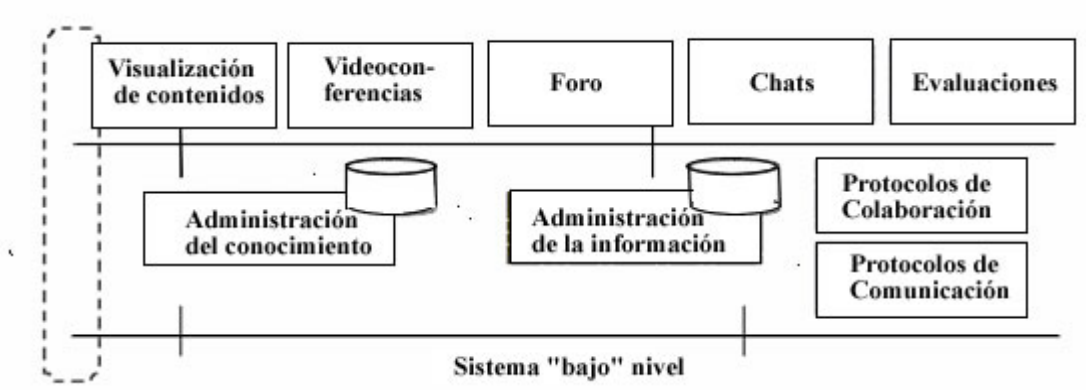

Figura 3: Arquitectura general de un de ambiente de aprendizaje virtual
Fuente: Hacia una infraestructura de componentes para la construcción de ambientes de aprendizaje colaborativo
Autor:Moreno Aguilar y Otros (2002)

Capa superior: se ubican las interfaces de usuario que representan servicios a los cuales éste tiene acceso. Dentro de los ambientes de aprendizaje colaborativo estos servicios son tanto de participación individual, como la visualización de contenidos y las evaluaciones así también como de colaboración tales como las videoconferencias, foros y chats.

Capa media: se encuentran diferentes protocolos y mecanismos tales como:

1 Protocolos de comunicación, que permiten el intercambio de mensajes con el sistema de bajo nivel para que éste, a su vez, intercambie mensajes con otras computadoras.
2 Protocolos de colaboración, que están soportados por comunicación y que permiten el intercambio de mensajes provenientes de la capa superior (de usuarios) con otros usuarios a través del sistema de bajo nivel. Los protocolos de comunicación deben considerar la colaboración tanto en forma síncrona como en forma asíncrona.
3 Mecanismos para la administración de la información que se encuentra en la capa inferior, es decir, almacenado y recuperación de datos.
4 Mecanismos para la administración del conocimiento, representado como información almacenada en la capa inferior. La forma general de búsqueda de información es a través de agentes.

Capa inferior: Se ubican los datos y su administración está representada por software de sistema de bajo nivel y es la encargada de comunicarse con la red.

Vista Estructural

Esta vista está compuesta por tres componentes:

1 Herramientas de administración del site

2 El conjunto de herramientas de interacción

3 Guía de estudios web

La herramienta de administración del site es un sistema de software basado en web que provee la funcionalidad necesaria para crear, manejar, actualizar y mantener el site, un ejemplo de este tipo de herramientas son Blackboard o WebCT.

El conjunto de herramientas de interacción, es la colección de medios que facilita la comunicación entre profesor y estudiante, entre ellos cabe mencionar: videoconferencia, correo electrónico y chats son algunos de los medios comúnmente usados para la interacción.

La guía de estudios web es el eje central del una herramienta e-learning, ya que este describe la estructura, el contenido, la interacción y los medios necesarios para soportar el proceso de enseñanza-aprendizaje, esta guía es también útil como mecanismo para soportar diferentes tipos de interacción entre estudiantes, profesores y contenidos, a continuación se presenta un ejemplo en UML de una guía de estudios web.

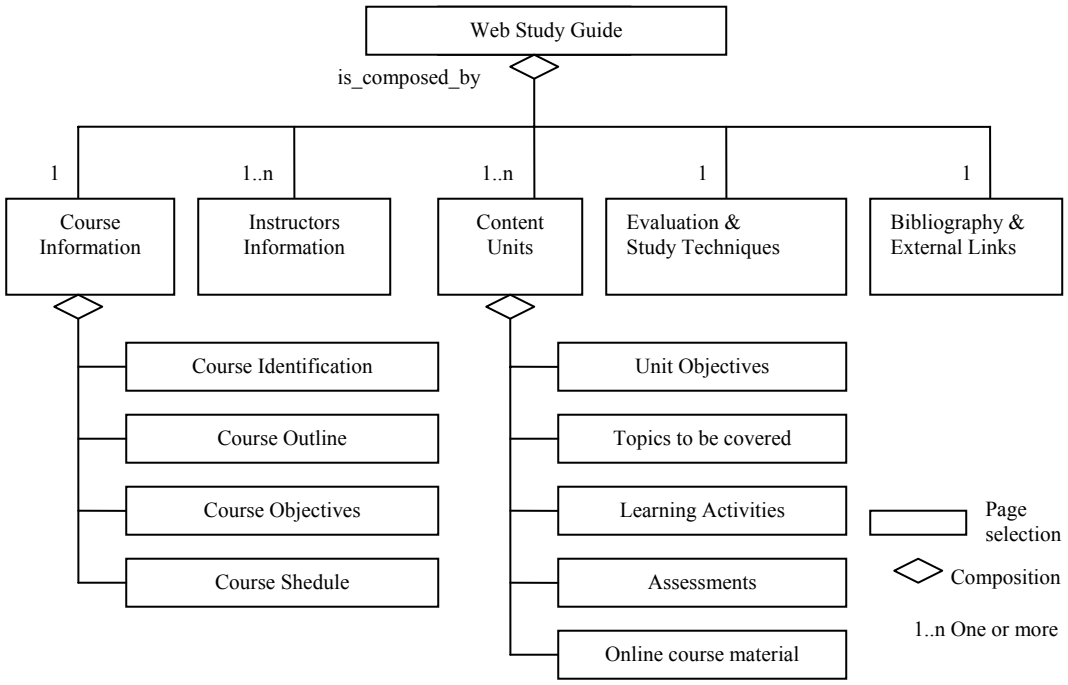

Figura 4: Estructura típica de una guía de estudios web

Fuente: A Software Engineering Approach to Manage the Development of Course Sites

Autor: Montilva y Sandia (2004)

Vista Funcional

Esta vista se concentra en las operaciones que están disponibles para los desarrolladores, profesores y alumnos.

Los desarrolladores deben tener la capacidad de administrar y mantener la herramienta de aprendizaje. Los profesores deben tener la capacidad para crear y actualizar el contenido de los cursos y monitorear a los estudiantes. Los alumnos deben tener acceso a los materiales explicativos de los conceptos de un dominio de conocimiento y a evaluaciones en línea que le presenten resultados inmediatos de la evaluación realizada para ello tienen asociado un esquema de navegación que permite

acceder a estos materiales e incluye mecanismos que permiten el acceso a asistencia guiada.

Definición de los estilos arquitecturales de software orientados a herramientas e-learning.

Arquitectura Orientada a Servicios

SOA "Arquitectura Orientada a Servicios", es una arquitectura cuyo propósito es lograr un débil acoplamiento entre los componentes de software que interactúan entre si. Débil acoplamiento es una condición en la que los elementos no están relacionados uno al otro, tal que pueden ser intercambiados fácilmente. Su objetivo es flexibilizar la interacción entre componentes. SOA no significa que deban exponerse como servicios todas las funcionalidades del sistema. Ort (2005)

La mayoría de las definiciones de SOA identifican la utilización de Web Service (empleando SOAP y WSDL) en su implementación, no obstante se puede implementar SOA utilizando cualquier tecnología basada en servicios. SOA se basa en el concepto ESB (Enterprise Service Bus), al que se conectan los servicios a través web services, siendo posible reutilizar componentes y procesos ya desplegados. En efecto, Gutiérrez y Otón (2004) expresan:

> "Realizando aplicaciones orientadas a servicio se pueden conectar aplicaciones heterogéneas con el aumento de flexibilidad que supone, y un punto muy importante es que permite que las organizaciones interactúen cuando realmente lo requieran, sin necesidad de tener conexiones permanentes. Como una arquitectura SOA se basa en estándares, el tiempo de aprendizaje de utilización de las tecnologías sobre las que se apoya se reduce drásticamente" (p.4)

En relación a esto, Sharma y Kitchens (2004) consideran que la Arquitectura Orientada a Servicios provee a un ambiente e-learning escalabilidad y una arquitectura abierta y global, proporcionando además flexibilidad para incrementar el nivel de sofisticación de la solución e-learning a medida que esta evoluciona En la figura 5 se muestra una propuesta de Arquitectura Orientada a Servicios para un ambiente de aprendizaje virtual

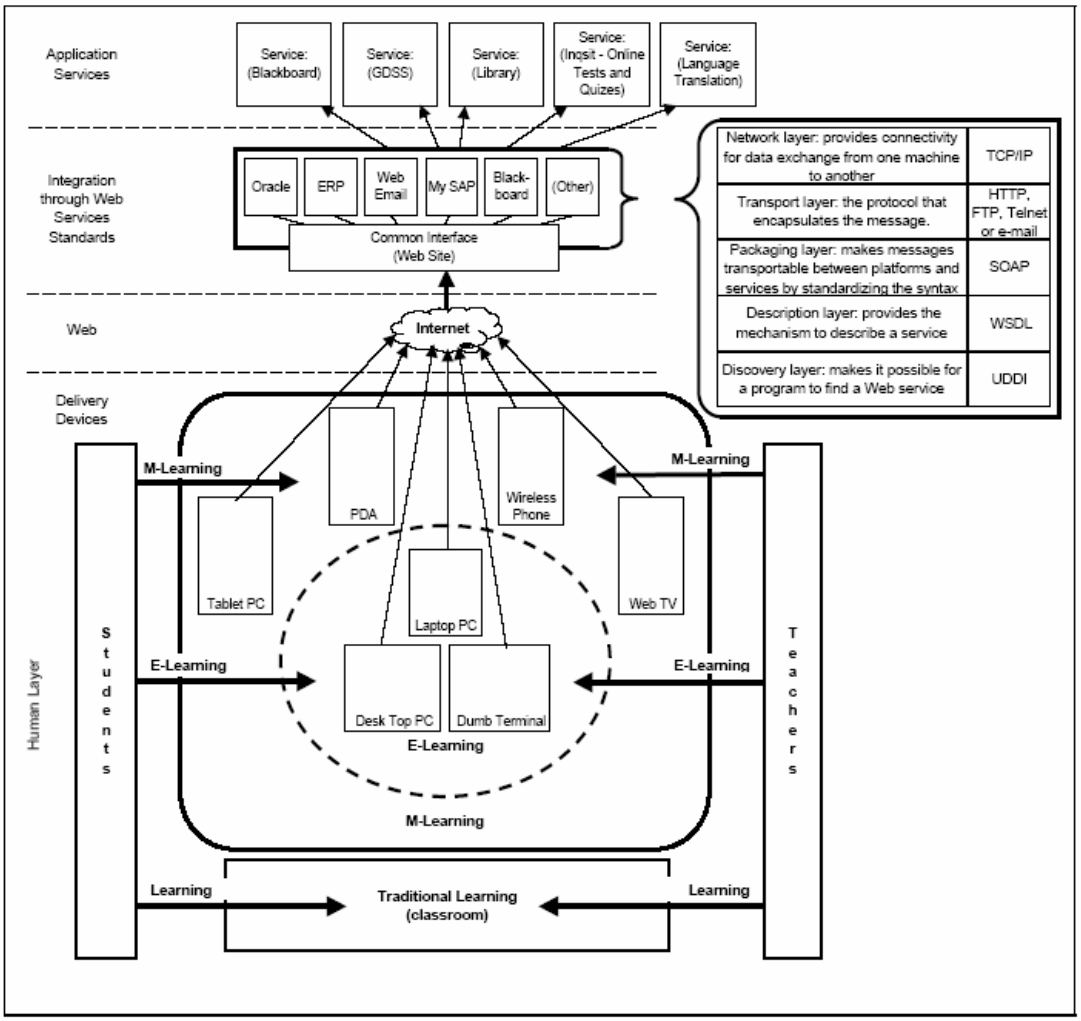

Figura 5: Arquitectura Orientada a Servicios para un e-learning

Fuente: Web Services Architecture for M-Learning

Autor: Sharma y Kitchens (2004)

Atributos de calidad presentes en una Arquitectura Orientada a Servicios.

Según Vojislav y otros (2004), la Arquitectura Orientada a Servicios puede evaluarse en los siguientes aspectos disponibilidad, seguridad y eficiencia. La disponibilidad y seguridad están presentes ya que este tipo de arquitectura provee garantía por encima de otras, por otro lado la eficiencia no es muy adecuada, debido a la gran cantidad de capas que posee Web Service, además de la necesidad de procesar estructuras de datos XML de esas capas, el procesamiento puede llegar a ser ineficiente, y este tipo de arquitectura llega a ser catalogada como de baja eficiencia y no adecuada para desarrollo de aplicaciones criticas.

Por su parte O'Brien y otros (2005), definen un conjunto de características de calidad asociados a la Arquitectura Orientada a Servicios como son alta fiabilidad, seguridad y eficiencia media en relación al rendimiento del sistema de software, en tanto que la eficiencia también se puede observar a otro nivel, el de la organización, ya que se transforman los procesos de negocio en servicios compartidos con un menor costo de mantenimiento.

Arquitectura Basada en Componentes

Un componente de software se define como una unidad de composición de aplicaciones software que posee un conjunto de requisitos, y que ha de poder ser desarrollado, adquirido, incorporado al sistema y compuesto con otros componentes, de forma independiente en tiempo y espacio. Szyperski (1998).

Por su parte, Montilva y Barrios (2004) expresan lo siguiente:

"El enfoque basado en componentes para el desarrollo de aplicaciones web tiene varias ventajas sobre otros enfoques. En primer lugar, el reuso

de componentes reduce los costos y tiempos de entrega debido a que la solución no es elaborada desde el comienzo, en segundo lugar, el enfoque basado en componentes promueve una mejor calidad debido al reuso de componentes ya probados".(p.329)

De manera general, un componente es un bloque de programa reusable que puede ser combinado con otros componentes en la misma computadora, o en otras computadoras pertenecientes a una red distribuida para construir una aplicación. Un componente implementa interfaces que son impuestas sobre él.

En relación a este tipo de arquitectura Montilva y Barrios (2004) proponen un modelo basado en componentes mediante el uso de arquitectura multi-capas denominado WATCH, el nombre deriva de las fases de desarrollo las cuales se llevan a cabo en el sentido de las agujas del reloj. El método consta de tres modelos, el modelo producto, el modelo de procesos y el modelo equipo.

Dado que la presente investigación se enfoca en el estudio de arquitecturas de software, se detalla a continuación el modelo producto el cual, es una descripción genérica del producto final utilizando el método, para ello este modelo es descrito en una vista que organiza los componentes de una aplicación web dentro de tres capas. La capa exterior trata con los aspectos de presentación de la aplicación, incluyendo las interfaces de usuario quien interactúa a través de esta capa. La capa media esta relacionada con la lógica del negocio de la aplicación, esta capa procesa los datos de entrada de datos y las transacciones o servicios requeridos por los usuarios a través de la capa exterior. La capa interna esta relacionada con la administración de los datos, esta capa almacena y devuelve los datos de una o más bases de datos. Una vista estructural de este modelo se describe en la Figura 6.

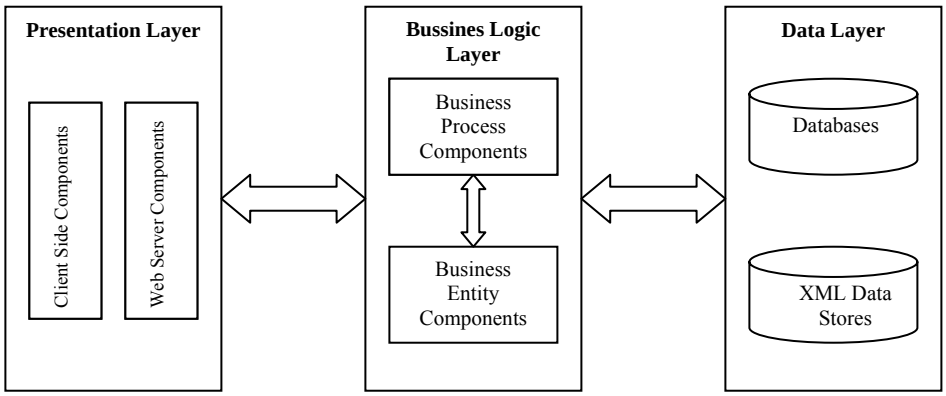

Figura 6: Modelo Producto Método WATCH

Fuente: The WATCH Method for Developing Web Applications.

Autor: Montilva y Barrios (2004)

Atributos de calidad presentes en una Arquitectura Basada en Componentes.

En estudio realizado por Bertoa y otros (2002) se evidencia que la mayoría de los atributos de tiempo de ejecución son muy difíciles de evaluar dado la ausencia casi total de información relativa a los mismos, a excepción del atributo Utilización de Recursos debido a que las necesidades de disco y memoria aparecen directamente en la página del proveedor, tal como se indica en la Tabla 9, otro atributo con un porcentaje bajo (10%) pero significativo en este contexto es el Tratamiento de errores dado que suele aparecer información relacionada en los manuales o guías.

Los atributos restantes que poseen un porcentaje pequeño pero superior a cero aparecen debido a que su propia funcionalidad esta relacionada con el atributo, por ejemplo, uno de los componentes analizados proporcionaba persistencia a otros componentes o aplicaciones.

Tabla 9: Evaluación de atributos de calidad en componentes de software presentes en tiempo de ejecución

Característica	Subcaracteristica	Atributo	Si(%)
Funcionalidad	Precisión	Precisión	0
		Exactitud Computacional	0
	Seguridad	Cifrado de Datos	5
		Capacidad de Control	0
		Capacidad para Auditar	5
Fiabilidad	Recuperabilidad	Secuenciable	0
		Persistente	5
		Transaccional	10
		Tratamiento de Errores	10
Eficiencia	Comportamiento Temporal	Tiempo de Respuesta	0
		Capacidad de Emisión	0
		Capacidad de Recepción	0
	Utilización de Recursos	Requisitos de Memoria	81
		Utilización de Disco	95

Fuente: Bertoa y otros (2002)

Determinación de características de calidad que debe poseer una Arquitectura de Software orientada a herramientas e-learning mediante encuesta diagnóstica.

Con el fin de presentar los resultados obtenidos en la aplicación de la encuesta diagnóstica a los profesores usuarios de la herramienta SABER de la UCLA se presenta a continuación para cada item o grupo de items la tabla de valores obtenidos expresados en frecuencia (f) y porcentajes (%). También se presenta el promedio ponderado y promedios porcentuales. El rango de los promedios ponderados utilizados para valorar la escala utilizada en el instrumento es el siguiente:

Tabla 10: Escala de valoración para promedios ponderados

Escala	Valor en el instrumento	Rango de valoración del promedio ponderado calculado
Indispensable	5	4.50 a 5.00
Sumamente importante	4	3.50 a 4.49
Medianamente importante	3	2.50 a 3.49
Poco importante	2	1.50 a 2.49
No se toma en cuenta	1	0.00 a 1.49

Fuente: La autora de la investigación

A continuación se muestran los resultados según las dimensiones que comprende la investigación

Dimensión: Funcionalidad

La primera dimensión del instrumento tuvo como finalidad diagnósticar la capacidad del software para proveer funciones que cumplan con las necesidades específicas o implícitas cuando este es utilizado, esta dimensión comprende dos sub características: Exactitud y Seguridad.

Sub característica: Exactitud

Esta sub-característica comprende los indicadores: Correcto, Adecuado y Precisión computacional.

De acuerdo a los datos mostrados en el Cuadro 1 se observa que el 67, 3% de los profesores encuestados indican que es indispensable la Exactitud en una

herramienta e-learning al igual el promedio ponderado apoya esta misma tendencia (4,65)

Cuadro 1
Exactitud

Valor	5		4		3		2		1		0						
Escala	Indisp.		Sum. Importante		Med. Importante		Poco Importante		No se toma en cuenta		No contestó		n		Prom. Ponderado	Indisp.+ Sum Importan.	
ITEM	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	∑ f	∑% ni	(∑f xvalor)/n	F	%ni
1. Libre de Fallas	16	61,5	10	38,5	0	0	0	0	0	0	0	0	26	100	4,61	26	100
2.Especificaciones	17	65,4	9	34,6	0	0	0	0	0	0	0	0	26	100	4,65	26	100
3. Personalizar	15	57,7	9	34,6	2	7,7	0	0	0	0	0	0	26	100	4,5	24	92,3
4. Precisión	22	84,6	4	15,4	0	0	0	0	0	0	0	0	26	100	4,84	26	100
Promedios		67,3		30,8		1,9		0		0		0			4,65		98
															Indispensable		

Fuente: La autora de la investigación

Sub característica: Seguridad

Esta sub-característica comprende los indicadores: Acceso y Auditoria. En relación a los resultados obtenidos en el Cuadro 2 se puede observar que el 78,8 % de los encuestados indican como indispensable y sumamente importante la seguridad en una herramienta e-learning.

Cuadro 2
Seguridad

Valor	5		4		3		2		1		0						
Escala	Indisp.		Sum. Importante		Med. Importante		Poco Importante		No se toma en cuenta		No contestó		n		Prom. Ponderado	Indisp.+ Sum Importan.	
ITEM	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	∑ f	∑% ni	(∑f xvalor)/n	f	%ni
5. Adm. Procesos	16	61,5	6	23	2	7,7	2	7,7	0	0	0	0	26	100	4,38	22	84,5
6. Auditoria	8	30,8	11	42,3	7	26,9	0	0	0	0	0	0	26	100	4,03	19	73,1
Promedios		46,2		32,6		17,3		3,9		0		0			4,20		78,8
															Sum. Importante		

Fuente: La autora de la investigación

Dimensión: Fiabilidad

Esta dimensión expresa la capacidad que tiene el software para reestablecer el nivel de ejecución y la capacidad de recuperación de datos cuando ocurre una falla.

Sub característica: Capacidad de recuperación

Los indicadores que comprende esta dimensión son: Tratamiento de errores, Comportamiento frente a la recuperación y Recuperación de datos. De acuerdo a los datos mostrados en el Cuadro 3 se observa que el 62,2 % de los profesores encuestados indican que es indispensable la Capacidad de recuperación en una herramienta e-learning al igual el promedio ponderado apoya esta misma tendencia (4,54).

Cuadro 3
Capacidad de recuperación

Valor	5		4		3		2		1		0						
Escala	Indisp.		Sum. Importante		Med. Importante		Poco Importante		No se toma en cuenta		No contesto		n		Prom. Ponderado	Indisp.+ Sum Importan.	
ITEM	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	∑ f	∑% ni	(∑f xvalor)/n	f	%ni
7.a. Detectar falla	17	65,4	8	30,8	1	3,8	0	0	0	0	0	0	26	100	4,61	25	96,2
7.b. Avisar falla	13	50	11	42,3	2	7,7	0	0	0	0	0	0	26	100	4,42	24	92,3
7.c. Notificar autom.	19	73,1	5	19,2	2	7,7	0	0	0	0	0	0	26	100	4,65	24	92,3
7.d. Funcionar adecuadamente	19	73,1	7	26,9	0	0	0	0	0	0	0	0	26	100	4,73	26	100
7.e Sin degradación	12	46,2	12	46,2	1	3,8	0	0	0	0	1	3,8	26	100	4,26	24	92,4
8.Respaldar datos	17	65,4	7	26,9	2	7,7	0	0	0	0	0	0	26	100	4,57	24	92,3
Promedios		62,2		32,1		5,1		0		0		0,6			4,54		94,2
															Indispensable		

Fuente: La autora de la investigación

Dimensión: Eficiencia

Esta dimensión expresa la capacidad del producto de software para proveer un desempeño apropiado en relación a la cantidad de recurso utilizado. Esta dimensión comprende las sub-características Comportamiento frente al tiempo y Uso de recursos.

Sub-característica: Comportamiento frente al tiempo

En el Cuadro 4 se puede observar como indispensable (4.72) disponer de rapidez en tiempo de respuesta para consultar y actualizar información, se observa que el 98,1 % de los encuestados consideran que la rapidez es indispensable y sumamente importante en una herramienta e-learning.

Cuadro 4
Comportamiento frente al tiempo

Valor	5		4		3		2		1		0						
Escala	Indisp.		Sum. Importante		Med. Important		Poco Importante		No se toma en cuenta		No contesto		n		Prom. Ponderado	Indisp.+ Sum Importan.	
ITEM	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	F	%ni	∑ f	∑% ni	(∑f xvalor)/n	f	%ni
9 a.Consultar inf.	21	80,8	4	15,4	1	3,8	0	0	0	0	0	0	26	100	4,76	25	96,2
9.b.Actualizar inf.	18	69,2	8	30,8	0	0	0	0	0	0	0	0	26	100	4,69	26	100
Promedios		75		23,1		1,9		0		0		0			4,72		98,1
															Indispensable		

Fuente: La autora de la investigación

Sub característica: Uso de recursos

En el Cuadro 5 se aprecia que el 43% de los encuestados consideran que es indispensable el uso de recursos tanto de espacio en disco como periféricos, se observa además que el 88,5% señala como indispensable que la herramienta

mantenga el mismo nivel de rendimiento cuando es usada simultáneamente por múltiples usuarios, en tanto que consideran el uso de periféricos como sumamente importante a excepción de dispositivo biométrico de reconocimiento de huellas dactilares como medianamente importante (2.80)

Cuadro 5
Uso de Recursos

Valor	5		4		3		2		1		0						
Escala	Indisp.		Sum. Importante		Med. Important		Poco Important		No se toma en cuenta		No contesto		n		Prom. Ponderado	Indisp.+ Sum Importante	
ITEM	F	%ni	f	%ni	f	%ni	f	%ni	f	%ni	f	%ni	∑ f	∑% ni	(∑f xvalor)/n	f	%ni
10. Espacio disco	15	57,7	10	38,5	0	0	0	0	0	0	1	3,8	26	100	4,42	25	96,2
11. Usuarios múltiples	23	88,5	3	11,5	0	0	0	0	0	0	0	0	26	100	4,88	26	100
12.a. Video Cámara	9	34,6	10	38,5	7	26,9	0	0	0	0	0	0	26	100	4,07	19	73,1
12. b. Micrófono	8	30,8	10	38,5	8	30,8	0	0	0	0	0	0	26	100	4	18	69,3
12. c. Audio	11	42,3	11	42,3	4	15,4	0	0	0	0	0	0	26	100	4,26	22	84,6
12.d. Huellas dactilares	1	3,8	5	19,2	10	38,5	8	30,8	2	7,7	0	0	26	100	2,80	6	23
Promedios		43		31,4		18,6		5,1		1,3		0,6			4,07		74,6
															Sumamente importante		

Fuente: La autora de la investigación

Conclusión fase diagnóstica

En el Gráfico 1 se resumen los promedios ponderados para cada una de las sub-características evaluadas, se puede observar que la sub-característica con mayor valoración es la de Comportamiento frente al tiempo (4,72), la cual involucra la rapidez que provee la herramienta para consultar y actualizar información. En tanto que sub-características como Exactitud (4,65) y Capacidad de Recuperación (4,54) tienen una valoración media por parte de los encuestados. Finalmente Seguridad (4,2) y Uso de Recursos (4,07) poseen baja valoración.

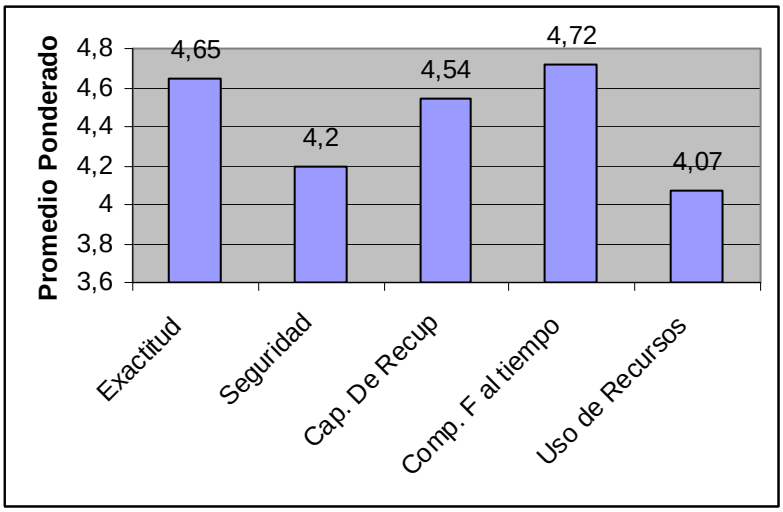

Gráfico 1. Tabla de frecuencia de atributos de calidad en tiempo de ejecución para herramientas e-learning.
Fuente: La autora de la investigación

Adicionalmente al diagnóstico realizado a través de la encuesta, no se observaron diferencias en la valoración dada por los profesores de acuerdo al Decanato al que pertenecen, manteniéndose la misma tendencia por estrato.

Análisis comparativo entre Arquitecturas Orientada a Servicios y Basada en Componentes.

Con el objeto de realizar el análisis comparativo, se tiene que, el objetivo del método SACAM es proporcionar un análisis racional en el proceso de selección de una arquitectura, comparando la aptitud de las arquitecturas candidatas a sistemas previstos. En la Figura 7 se desglosan los pasos que comprende el método:

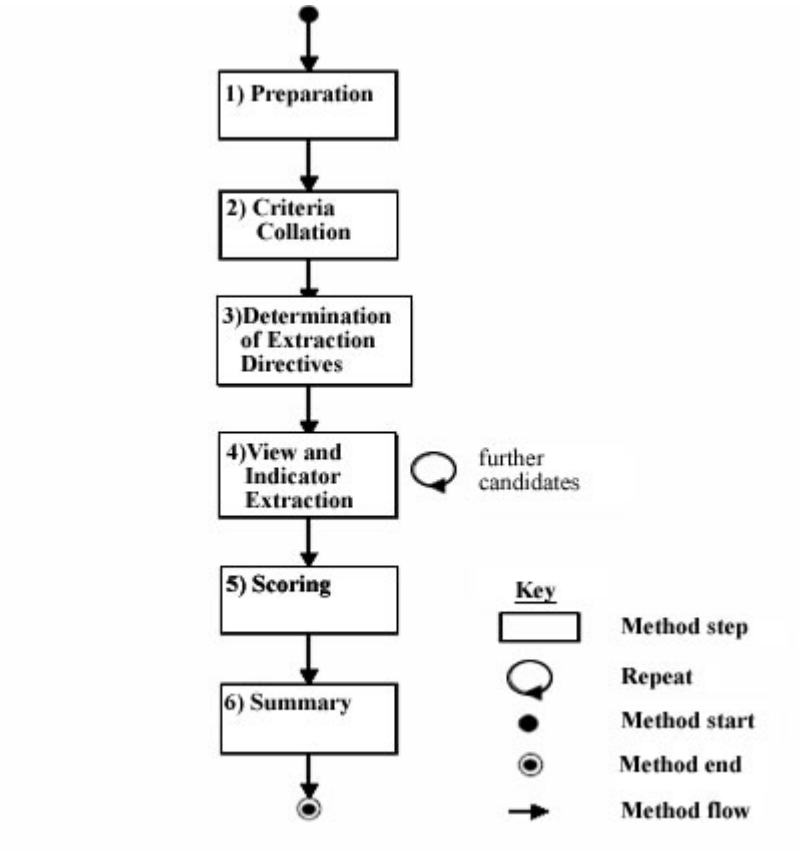

Figura 7. Pasos del método SACAM

Fuente: SACAM: The Software Architecture Comparison Analysis

Autor: Stoermer y otros (2003)

- Preparación: identifica las metas de negocio relevantes necesarias en la comparación y examina la documentación disponible para cada candidato de la arquitectura.
- Selección de los criterios: deriva los criterios de la comparación de las metas de negocio y las refina a ellos en escenarios con atributos de calidad
- Determinación de los directorios de la extracción, determina las visiones, la táctica, los estilos, y los patrones arquitectónicos que se buscan durante las extracciones siguientes para encontrar la evidencia de soporte para los escenarios del paso anterior.

- Extracción de la visión y de los indicadores, extrae las visiones arquitectónicas para cada candidato según los directorios de la extracción del paso anterior.
- Detecta los indicadores que apoyan los escenarios de los atributos de calidad del paso de selección de criterios.
- Se resumen los resultados del análisis y se proporciona una recomendación para el procedimiento de toma de decisión.

En los que respecta a los pasos iniciales del método SACAM como la preparación y selección de criterios estos han sido desarrollados como parte del presente trabajo de investigación. Se analizaron los componentes que integran una herramienta e-learning y se seleccionaron criterios de calidad adaptados a este tipo de herramientas.

Adicionalmente se definieron las arquitecturas existentes y se seleccionaron las arquitecturas candidatas de acuerdo al entorno estudiado; se detectaron los indicadores que apoyan los atributos de calidad en las dos herramientas seleccionadas según los resultados de la encuesta diagnóstica; complementando de esta manera los tres pasos siguientes del método. Finalmente, como resultado del análisis, se resume que la Arquitectura Orientada a Servicios en relación con la Arquitectura Basada en Componentes posee algunas ventajas para su aplicación en herramientas e-learning las cuales se detallan a continuación.

Según los resultados de la fase diagnóstica, la sub-característica Capacidad de recuperación perteneciente al atributo Fiabilidad tiene una valoración media, esta sub-característica tiene un alto grado en comparación con la Arquitectura Basada en Componentes.

En relación a la sub-característica Comportamiento frente al tiempo, perteneciente al atributo Eficiencia, la Arquitectura Basada en Componentes no

ofrece información en relación al grado de rendimiento, en tanto la Arquitectura Orientada a Servicios posee una valoración media y su aplicación esta orientada a sistemas no críticos.

Por ultimo es conveniente notar que la Arquitectura Basada en Componentes posee atributos de calidad variable dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo. La Arquitectura Orientada a Servicios fomenta los estándares y protocolos que hacen más fácil acceder a su contenido y entender su funcionamiento, por esto es mayor su flexibilidad y calidad en comparación con la Arquitectura Basada en Componentes.

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Luego de realizada la presente investigación se llegó a las siguientes conclusiones que dan respuesta a las interrogantes en el estudio:

1. Una Arquitectura de Software para herramientas e-learning debe proveer un buen desempeño al ser usado bajo entorno web.
2. En el estudio diagnóstico realizado se evidencia la importancia de disponer rapidez en la velocidad de actualización y consulta de la información presente en una herramienta e-learning; en tanto que sub-características como Exactitud y Capacidad de recuperación tienen una valoración media. Seguridad y Uso de Recursos poseen baja valoración.
3. No se observaron diferencias en la valoración dada por los profesores de acuerdo al Decanato al que pertenecen, conservándose la misma tendencia por estrato de la muestra seleccionada.
4. Como resultado de la comparación de las Arquitecturas Orientadas a Servicios y Basada en Componentes se resume que la Arquitectura Orientada a Servicios en relación con la Arquitectura Basada en Componentes posee algunas ventajas para su aplicación en herramientas e-learning. Entre las cuales se encuentran:

    - La sub-característica Capacidad de recuperación perteneciente al atributo Fiabilidad tiene una valoración media según los resultados de la

fase diagnóstica, esta sub-característica tiene un alto grado en la Arquitectura Orientada a Servicios en comparación con la Arquitectura Basada en Componentes.

- En la sub-característica Comportamiento frente al tiempo, perteneciente al atributo Eficiencia, la Arquitectura Basada en Componentes no ofrece información en relación al grado de rendimiento, en tanto la Arquitectura Orientada a Servicios posee una valoración media y su aplicación esta orientada a sistemas no críticos.

- Finalmente, la Arquitectura Basada en Componentes posee atributos de calidad variable dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo. La Arquitectura Orientada a Servicios fomenta los estándares y protocolos que hacen más fácil acceder a su contenido y entender su funcionamiento, por esto es mayor su flexibilidad y calidad en comparación con la Arquitectura Basada en Componentes.

Recomendaciones

Tomando en cuenta las conclusiones y los hallazgos planteados, se considera conveniente:

1. Extender el estudio a los atributos de calidad presentes en el ciclo de vida, como Interoperabilidad, Madurez y Mantenibilidad el cual proporcionará una mayor apreciación sobre la Arquitectura de Software.

2. Evaluar la incorporación de dispositivos móviles y conocer su impacto en el uso de herramientas e-learning.
3. Valorar los atributos de calidad que debe poseer una herramienta e-learning desde el enfoque del usuario Estudiante.
4. Finalmente, extender el estudio a otras Arquitecturas de software orientadas hacia herramientas e-learning.

GLOSARIO DE TERMINOS

ADL: (Advanced Distributed Learning Network). Iniciativa del Departamento de defensa estadounidense para conseguir interoperabilidad entre computadores y software de aprendizaje basado en Internet, a través del desarrollo de un marco técnico común que almacena el contenido en forma de objetos de aprendizaje reutilizables.

API: (Aplication Program Interface) Interfaz para programas de aplicación. Conjunto de convenciones de programación que definen cómo se invoca un servicio desde un programa.

CBT: (Computer Based Training). Formación basada en computador. Curso o material educativo presentado por computador, generalmente mediante CD ROM o disco flexible. A diferencia de la formación on line, no requiere que el computador esté conectado a la red y generalmente no tiene enlaces a recursos externos al curso.

CMI: (Computer Managed Instruction). Uso del computador para administrar procesos de aprendizaje.

CMS: (Content Management System). Sistema de gestión de contenidos. Aplicación de software que simplifica la creación y administración de contenidos por medio de páginas web.

CORBA: (Common Object Request Broker Architecture). Es un estándar que establece una plataforma de desarrollo de sistemas distribuidos facilitando la invocación de métodos remotos bajo un paradigma orientado a objetos.

Courseware: Cualquier programa de software de tipo instruccional o educacional.

DCOM: Modelo de objetos componentes distribuido. Es una extensión del modelo de objetos componentes (COM), que facilita la distribución transparente de objetos a través de la red e Internet..

Estándar: Unidad de medida adoptada y aceptada comúnmente como criterio.

Encapsulación: Es el criterio que agrupa en una clase todas las variables que determinan el estado de un objeto y los métodos que permiten acceder a ellas.

Framework: Es una estructura de soporte definida, en la cual un proyecto de software puede ser organizado y desarrollado.

Herencia: Medio por el cual se puede construir un objeto a partir de otro.

IEEE: (Institute of Electrical and Electronics Engineers). Instituto de Ingenieros Eléctricos y Electrónicos (USA).

Inteligencia Artificial: Conjunto de técnicas e investigaciones relacionadas con la posibilidad de que una máquina pueda simular los procesos de razonamiento que caracterizan al cerebro humano.

Intranet: Red de uso privado que emplea los mismos estándares y herramientas de Internet.

IMS: (Instructional Management System). Sistema de gestión Instruccional, Consorcio de aprendizaje global. Coalición de organizaciones gubernamentales dedicadas a definir y distribuir especificaciones de interoperabilidad de arquitectura abierta para productos de e-learning.

ISO: (International Standard Organization). Organización de estándares Internacionales.

LMS: (Learning Management System). Software que automatiza la administración de acciones de formación. Un LMS registra usuarios, organiza los diferentes cursos en un catálogo, almacena datos sobre los usuarios, también provee informes para la gestión. Un LMS es diseñado generalmente para ser utilizado por diferentes editores y proveedores. Generalmente no incluye posibilidades de autoría (creación de cursos propios), en su lugar, se centra en gestionar cursos creados por gran variedad de fuentes diferentes. Generalmente también se le conoce como plataforma.

LTSC: (Learning Technologies Standards Committee). Comité de la IEEE que tiene por objetivo desarrollar estándares técnicos, prácticas recomendadas y guías para la implementación informática de sistemas de formación a distancia.

LO: (Learning Object). Objetos de aprendizaje: Unidad reusable de información independiente de los medios. Bloque modular de contenido para e-learning**.**

Metadato: Datos sobre los datos, esto es información sobre la información misma.

Método: Son procedimientos que un objeto puede ejecutar.

Modelo: Un modelo es una conceptualización de un evento, un proyecto, una hipótesis, el estado de una cuestión, que se representa como un esquema con símbolos descriptivos de características y relaciones más importantes

Objeto: Un objeto es una instancia de una clase, que encapsula estado y procesos.

On-line: En línea. Estado en el que un computador está conectado a otro computador o servidor a través de una red.

Polimorfismo: En programación orientada a objetos se denomina polimorfismo a la capacidad del código de un programa para ser utilizado con diferentes tipos de datos u objetos

Realidad Virtual: Es la simulación de un entorno real o imaginario que puede ser experimentado visualmente en las tres dimensiones el alto, el ancho y la profundidad , además puede proveer una experiencia visual interactiva en tiempo completo, ya sea con sonido o con movimiento y posiblemente con el tacto y otras formas de retroalimentación

SCORM: (Shareable Courseware Object Reference Model). Resultado de la iniciativa de Aprendizaje avanzado distribuido (ADL) del Departamento de Defensa Estadounidense. Los elementos de la plataforma de SCORM pueden ser combinados fácilmente con otros elementos compatibles para producir reposiciones altamente modulares de materiales de formación.

SOAP (Simple Object Access Protocol). Es un dialecto de XML el cual permite a las aplicaciones invocar métodos de objetos remotos, así como recibir las respuestas de los mismos.

Web Semántica: La Web semántica tiene como objetivo crear un medio universal para el intercambio de información basado en representaciones del significado de los recursos de la Web, de una manera inteligible para las máquinas. Con ello se pretende ampliar la interoperabilidad entre los sistemas informáticos y reducir la mediación de operadores humanos en los procesos inteligentes de flujo de información

WBT (Web Based Training): Formación basada en la Web. Provisión de contenido educativo a través de un navegador web, ya sea en Internet, en una intranet privada o una extranet.

WSDL (Web Service Description Language). Es al igual que SOAP, un dialecto de XML que contiene información acerca de la interfaz, semántica y administración de una llamada a un servicio Web.

UDDI (Universal, Description, Discovery, and Integration). Es un protocolo para describir los componentes disponibles de servicios Web.

UML (Unified Modelling Language): Lenguaje de modelado de sistemas.

XML (Extensible Markup Language): Lenguaje de codificación de última generación, que permite a los diseñadores Web programar sus propios comandos de marcación. Estos comandos podrán ser usados posteriormente como si fueran comandos HTML estándares.

REFERENCIAS BIBLIOGRAFICAS

Anido, Luis E.y otros. 2002. A Step ahead in E-learning Standardization: Building Learning Systems from Reusable and Interoperable Software Components.Universidad de Vigo. España. URL: http://www2002.org/CDROM/alternate/136/ (Consulta: Noviembre 10, -2005)

Angulo, A y otros. 2002. Evaluación de Arquitecturas de Software para un KMS bajo un enfoque de calidad. Universidad de Carabobo Facultad de Ciencias y Tecnología y Universidad Simón Bolívar.Venezuela.

Bass, Len y otros. 1998. Software Architecture in Practice. Reading, Addison-Wesley.

Bertoa y otros.2002. Atributos de calidad para componentes COTS. Universidad de Malaga, España. URL: www.lcc.uma.es/~av/Publicaciones/03/TICS03.pdf (Consulta: Enero 09, 2006)

Booch, G. 1998. Software Architecture and the UML. URL:www.rational.com/uml (Consulta: Noviembre 10, 2005)

Burbeck, Steve 1992. Application programming in Smalltalk-80: How to use Model-View-Controller (MVC). University of Illinois in Urbana-Champaign, Smalltalk Archive. URL: http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html. (Consulta: Noviembre 21, 2005)

Clements. Paul. 1996 A Survey of Architecture Description Languages. Proceedings of the International Workshop on Software Specification and Design, Alemania.

Cronbach, L. 1951. Coefficient alpha and the internal structure of test. Psychometrika.Vol 16.

Deming, W. 1989. Calidad, productividad y competitividad, la salida a la crisis. Diaz de Santos. Madrid.

Doberkat, Ernst 2002. Pipes and filters: Modelling a software architecture through relations. Internes Memorandum des Fachbereich Informatik Lehrstuhl für Software Technologie, Universidad de Dortmund, Memo N° 123. URL: ftp://ls10-ftp.cs.uni-dortmund.de/pub/Technische-Berichte/Doberkat_SWT-Memo-123.pdf (Consulta: Noviembre 21, 2005)

Fielding Roy Thomas. 2000.Architectural styles and the design of network-based software architectures. Tesis doctoral, University of California, Irvine.

Foix y Zavando.2002. Estandares e-learning. Estado del Arte. Centro de Tecnologías de Información Intec. Chile.

Gamma Erich y otros. 1995. Design Patterns: Elements of reusable object-oriented software. Reading, Addison-Wesley.

Garlan, David.2000. Software Architecture: A Roadmap. En Anthony Finkelstein. The future of software engineering, ACM Press.

Garlan, David y Shaw, Mary. 1994. An introduction to software architecture. CMU Software Engineering Institute Technical Report, CMU/SEI-94-TR-21, ESC-TR-94-21.

Gutiérrez Isaac y Salvador Otón. 2004. Arquitecturas Orientadas a Servicios. Universidad de Alcalá. España.

Hernández, Sampieri y otros. 2003. Metodología de la Investigación. Tercera Edición. Mc Graw Hill.

IEEE. 2000. URL: www.ieee.org. (Consulta: Noviembre 15, 2005)

ISO/IEC 14598-5. 1998. Information Technology- Software product evaluation- Part: 5 Process for evaluation ISO/IEC Organization.

ISO/IEC 9126. 2001. Information Technology- Software product evaluation- Part:5 Process for evaluators for their uses. ISO/IEC Organization.

Juran, J. 1990. Juran y el liderazgo para la calidad. Diaz de Santos.

Juristo, N. 2002. Master en Ingenieria del Software e IC. Facultad de Informatica Universidad Politecnica de Madrid. URL: http://www.ls.fi.upm.es/udis/miembros/natalia (Consulta: Noviembre10, 2005)

Klein, Mark y Kazman, Rick. 1999. Attribute-based architectural styles. Technical Report, CMU/SEI-99-TR-022, ESC-TR-99-022, Carnegie Mellon University.

Losavio, Francisca y otros. 2001. Quality Models to Design Software Architectures. URL: http://doi.ieeecomputersociety.org/10.1109/TOOLS.2001.911761 (Consulta: Noviembre 15,2005)

Losavio, Francisca y otros.2004.Designing Quality Architecture: incorporating ISO Standards into the Unified Process. Systems Management. Completar ciudad

Moneva, M.C. 2003. El futuro del e-learning: el futuro de la educación. Proyecto SI-Loc@l. Oviedo, España.

Montilva y Barrios. 2004. The WATCH Method for Developing Web Applications. Sistemas de Información e Ingeniería de Sofware: Temas Selectos. Centro de Estudios en Informática. Mérida. Venezuela

Montilva y Sandia. A Software Engineering Approach to Manage the Development of Course Sites. Sistemas de Información e Ingenieria de Sofware: Temas Selectos. Centro de Estudios en Informática. Merida. Venezuela.

Moreno Aguilar y otros. 2002. Hacia una infraestructura de componentes para la construcción de ambientes de aprendizaje colaborativo. 2002.
URL: http://bibliotecadigital.conevyt.org.mx/colecciones/documentos/somece/59.pdf (Consulta Enero 24, 2006).

Multiple Instruction Multiple Data.(MIMD).URL:http://en.wikipedia.org/wiki/MIMD

Muñoz, Alvaro. 2005. Usabilidad del Sistema de Aprendizaje Basado en Redes (SABER) para Estudiantes de la Materia Multimedia en una Universidad Venezolana.

O'Brien, Liam y otros. 2005. Quality Attributes and Service-Oriented Architectures.
URL:http://www.sei.cmu.edu/publications/documents/05.reports/05tn014/05tn014.html#chap04. (Consulta Marzo 24, 2006)

Ort, Ed. 2005. Service-Oriented Architecture and Web Services: Concepts, Technologies, and Tools. URL: http://java.sun.com/developer/technicalArticles/WebServices/soa2/soa2.pdf (Consulta Marzo 24, 2006)

Otero, Juan Carlos. 2001. Introducción al e-learning. UCV. Caracas, Venezuela

Perry Dewayne y Wolf Alexander L. 1992.Foundations for the study of software architecture. ACM SIGSOFT Software Engineering Notes, 17(4), pp. 40–52.

Preiss y otros. 2001. On Quality Attribute Based Software Engineering".27 Euromicro Conference, Warsaw, Polonia, Septiembre 2001. IEEE CS Press.

Pressman R. 2002. Ingenieria del Software. Un enfoque práctico. Mc Graw-Hill. Quinta Edición.

Rosenberg, M. 2001. E-learning: Estrategias para transmitir conocimiento en la era digital. Bogotá. McGraw-Hill Interamericana.

Sharma y Kitchens.2004. Web Services Architecture for M-Learning. University Ball State. USA. Electronic Journal on e-Learning Volume 2 Issue 1.

Shaw, Mary y Garlan, David.1996.Software Architecture: Perspectives on an emerging discipline. Upper Saddle River, Prentice Hall

Silvio, José y Marié Lapierre(2003) . Cooperación Universidad-Empresa en el E-Learning: experiencias en Iberoamérica. Instituto Internacional de la UNESCO para la Educación Superior en América Latina y el Caribe- IESALC

Single Instruction Multiple Data. (SIMD).URL:http://en.wikipedia.org/wiki/SIMD

Schmitt, Peter. 2005. Mobile Communication Solution Data Acquisition. URL: http://www.isc-soft.de/isc-temp- page/Mobile%20communication/Mobile%20communication%20overview%20EN.pdf. (Consulta: Diciembre 01, 2005)

Staff High Editores. 2003. Tecnología y Gobierno. URL:http://infochannel.com.mx (Consulta: Noviembre 21,2005)

Stoermer y otros. 2003. SACAM: The Software Architecture Comparison Analysis Method. URL: http://www.sei.cmu.edu/publications/documents/03.reports/03tr006/03tr006.html#chap01. (Consulta Noviembre 14, 2005)

Szyperski, Clemens. 1995. Component Oriented Programming: A refined variation of Object-Oriented Programming. The Oberon Tribune.

Tapia Machain, Ana Mercedes. Instituciones de educación a distancia en América Latina. URL: http://www.uned.es//catedraunesco-ead/anatapia/indice.htm (Consulta Diciembre 01, 2005)

Universidad Centroccidental "Lisandro Alvarado". 2002. Manual para la presentación del Trabajo Conducente al Grado Académico de: Especialización – Maestría- Doctorado.

Universidad Centroccidental "Lisandro Alvarado". 2005. URL: http://www.ucla.edu.ve/valores/mision.htm (Consulta Noviembre 14, 2005)

Vojislat y otros .2004. A Quality Evaluation Framework For Web Service Architectures. University of Manitota. Canada. URL: www.cs.umanitoba.ca/~softart/papers/mcetech05.pdf (Consulta: Marzo 07,2006)

Xiaofei Liu y otros.2003. An implementable architecture of an e-learning. Universidad de Otawa.

Xiaohong Qiu y otros.2003. Web Service Architecture for E-learning. University of Indiana.

Wang, Guijun y Fung Casey. 2004. Architecture paradigms and their influences and impacts on componente-based software systems. Proceedings of the 37th Hawaii International Conference on System Sciences.

Westerkamp, Peter. 2003.E-Learning as Web-Service. University of Munster. Alemania.URL:http://www.dbs.cs.uniduesseldorf.de/gvd2004/papers/WesterkampPeter.pdf (Consulta: Noviembre 14,2005)

ANEXOS

ANEXO 1

Cuestionario aplicado a docentes usuarios de la herramienta SABER de la Universidad Centroccidental "Lisandro Alvarado"

ENCUESTA DIAGNOSTICA SOBRE CALIDAD EN ARQUITECTURA DE SOFTWARE ORIENTADA A HERRAMIENTAS E-LEARNING

Decanato al que pertenece:

	Administración y Contaduría		Ciencias y Tecnología		Medicina
	Agronomía		Ingeniería Civil		Veterinaria

A continuación se le presenta una serie de planteamientos con cinco (5) alternativas de respuesta, las cuales tienen como finalidad determinar las características de calidad asociadas a la Arquitectura de Software, orientadas a herramientas e-learning o de aprendizaje a distancia. Debe considerar la siguiente escala:

Indispensable	Sumamente importante	Medianamente importante	Poco importante	No se toma en cuenta
(5)	(4)	(3)	(2)	(1)

En relación a los siguientes planteamientos indique **el nivel de apreciación necesario que debe estar presente en una herramienta e-learning**. Seleccione sólo una de ellas, marcando con una (X) la casilla que mejor se ajuste a su apreciación.

OPINION SOBRE FUNCIONALIDAD EN HERRAMIENTAS E-LEARNING	(5)	(4)	(3)	(2)	(1)
1. Provea procesos o funcionalidades libre de fallas					
2. Las funcionalidades satisfagan las especificaciones requeridas por el usuario					
3. Permita personalizar el entorno de la herramienta para ajustarlo a las necesidades del Usuario					
4. La herramienta sea precisa en el manejo de la información					
5. Permita al profesor administrar los procesos a los cuales los estudiantes pueden acceder					
6. Disponga de un mecanismo de auditoria que permita registrar las operaciones que llevan a cabo los usuarios					

OPINION SOBRE FIABILIDAD EN HERRAMIENTAS E-LEARNING	(5)	(4)	(3)	(2)	(1)
7. En caso de presentarse una falla:					
a. La herramienta debe ser capaz de detectar la falla					
b. Avisar de la falla mediante un mensaje					
c. Notificar automáticamente la falla al administrador					
d. Los procesos no asociados a la falla deben funcionar adecuadamente					
e. Los procesos deben funcionar sin degradación en relación a su velocidad de ejecución					
8. Provea la funcionalidad de respaldar datos					

OPINION SOBRE EFICIENCIA EN HERRAMIENTAS E-LEARNING	**(5)**	**(4)**	**(3)**	**(2)**	**(1)**
9. Provea rapidez en tiempo de respuesta para :					
a. Consultar información					
b. Actualizar información					
10. Disponga de espacio en disco acorde al promedio de almacenamiento usado por los profesores					
11. Mantenga el mismo nivel de rendimiento cuando la herramienta es utilizada simultáneamente por múltiples usuarios					
12. La herramienta permita el uso de los siguientes periféricos:					
a. Video Cámara					
b. Micrófono					
c. Dispositivos de Audio					
d. Dispositivo de reconocimiento de huellas dactilares.					

ANEXO 2

Carta de Solicitud de Validación a los Expertos

UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"
DECANATO DE CIENCIAS Y TECNOLOGIA

Barquisimeto, Febrero de 2006

Ciudadano(a): _____

Reciba un cordial saludo en mi nombre. Sirva la presente para solicitar ante Ud. su colaboración en calidad de experto para determinar la validez de contenido del cuestionario con el cual se pretende recabar la información necesaria para el Trabajo de Grado titulado "Estudio de estilos en la arquitectura del software orientados a herramientas e-learning utilizando en método SACAM" el cual será aplicado a los docentes activos usuarios de la herramienta SABER de la UCLA.

Para tal propósito se anexa a la presente 1) Matriz de Operacionalización de las Variables; 2) Formatos de Validación con sus respectivas instrucciones; 3) Cuestionarios que se aplicarán a la muestra.

Sin otro particular al cual hacer referencia y agradeciendo de antemano su colaboración, queda de Ud.

Atentamente,

_____

Ing. María Elena Torres Samuel
C.I: 11.791.306

ANEXO 3

Formato de Validación del Instrumento

UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"
DECANATO DE CIENCIAS Y TECNOLOGIA
POSTGRADO EN CIENCIAS DE LA COMPUTACION
MENCION INGENIERIA DEL SOFTWARE

ESTUDIO DE ESTILOS EN LA ARQUITECTURA DEL SOFTWARE ORIENTADOS A HERRAMIENTAS E-LEARNING UTILIZANDO EL METODO SACAM

Para efectos de la evaluación correspondiente a los ítems del instrumento a ser utilizado en la tesis. Se agradece revisar cada uno de los planteamientos de acuerdo a los siguientes criterios.

a) **Pertinencia**: Es la correspondencia del item con el aspecto
b) **Claridad**: Se refiere a la redacción precisa y sencilla del item
c) **Congruencia**: Entendida como la lógica interna del item

Se le agradece seleccionar una d las 2 posibles opciones (Si/No) para cada ítems con el objetivo de señalar el grado de pertinencia, claridad y congruencia de los ítems.

	Pertinencia		Claridad		Congruencia		Observaciones
Item	SI	NO	SI	NO	SI	NO	
1	☐	☐	☐	☐	☐	☐	_____
2	☐	☐	☐	☐	☐	☐	_____
3	☐	☐	☐	☐	☐	☐	_____
4	☐	☐	☐	☐	☐	☐	_____
5	☐	☐	☐	☐	☐	☐	_____
6	☐	☐	☐	☐	☐	☐	_____
7 a	☐	☐	☐	☐	☐	☐	_____
7 b	☐	☐	☐	☐	☐	☐	_____
7 c	☐	☐	☐	☐	☐	☐	_____
7 d	☐	☐	☐	☐	☐	☐	_____

	Pertinencia		Claridad		Congruencia		Observaciones
Item	SI	NO	SI	NO	SI	NO	
7 e	☐	☐	☐	☐	☐	☐	_____
8	☐	☐	☐	☐	☐	☐	_____
9 a	☐	☐	☐	☐	☐	☐	_____
9 b	☐	☐	☐	☐	☐	☐	_____
10	☐	☐	☐	☐	☐	☐	_____
11	☐	☐	☐	☐	☐	☐	_____
12 a	☐	☐	☐	☐	☐	☐	_____
12 b	☐	☐	☐	☐	☐	☐	_____
12 c	☐	☐	☐	☐	☐	☐	_____
12 d	☐	☐	☐	☐	☐	☐	_____

A ResumenCurricular del Autor

María Elena Torres Samuel, Portadora de la C.I: V- 11791306, nace en Barquisimeto el 05 de Octubre de 1975. Obtiene el titulo de bachiller en el Liceo "Ezequiel Bujanda" de Barquisimeto. Ingresa a la Universidad Centroccidental " Lisandro Alvarado" de Barquisimeto en el año 1993, para iniciar estudios de pregrado en la carrera Ingeniería en Informática. En el año 1999 recibe el titulo de Ingeniero en Informática. Durante el año 2000 se desempeña como Analista de Soporte Interno en la empresa Softech Sistemas C.A, en Caracas D.C. Posteriormente durante los años 2001 y 2002 se desempeña como Analista Programador en la empresa QoS2000 C.A. Durante los años 2003 a la actualidad se desempeña como Analista de Sistemas en la empresa Milo C.A.ubicada en Barquisimeto.

Tabla 6:

Operacionalización de la variable en estudio

Objetivo	Variable	Dimensiones	Sub-caracteristicas	Indicadores	Item
Propósito Determinar las características de calidad que debe poseer una Arquitectura del Software orientada a e-learning. **Definición** Una característica de calidad se define como un conjunto de atributos para la cual la calidad es descrita y evaluada.	Características de calidad para Arquitectura de software **en tiempo de ejecución**	**Funcionalidad:** Es la capacidad para proveer funciones que cumplan con las necesidades especificas o implícitas cuando el software es usado bajo ciertas condiciones	**Exactitud:** Habilidad para proveer los resultados correctos o adecuados con el grado de precisión necesario	**Correcto**: Libre de errores o defectos.	1
				Adecuado: Apropiado a las condiciones requeridas por el usuario.	2,3
				Precisión Computacional: Certidumbre y conveniencia exacta en los cálculos computacionales.	4
			Seguridad: Es la habilidad para resistir usos no autorizados.	**Acceso:** Mecanismo que permite administrar los procesos o funcionalidades que el usuario puede utilizar.	5
				Auditoria: Mecanismo que permite registrar la ejecución de los procesos de usuario.	6
		Fiabilidad: Es la capacidad para mantener un nivel especifico de rendimiento.	**Capacidad de recuperación:** Es la capacidad para reestablecer el nivel de ejecución, la capacidad para recuperar los datos y el tiempo y esfuerzo necesario para hacerlo.	**Tratamiento de errores:** Indica si se realiza algún tipo de acción cuando una falla ocurre.	7 a, 7 b, 7 c
				Comportamiento frente a la recuperación: Forma en la cual el sistema actúa cuando ocurre una falla, en relación a la degradación de recursos.	7 d, 7 e
				Recuperación de datos: Mecanismo que permite la accesibilidad a los datos después de una falla.	8
		Eficiencia: Es la capacidad del producto de software para proveer un desempeño apropiado relativo a la cantidad de recurso utilizados	**Comportamiento frente al tiempo**: Es la capacidad del producto de software de proveer tiempos de respuesta y tiempos de procesamiento apropiados	**Tiempo de respuesta en recuperar información**: Tiempo necesario para mostrar los datos requeridos	9 a
				Tiempo de respuesta en actualizar información: Tiempo necesario para actualizar información en el sistema.	9 b
			Uso de recursos: Cantidad y tipo de recursos usados en la ejecución de las funciones	**Uso de memoria:** Utilización de dispositivos físicos de almacenamiento para manejar el sistema de software	10,11
				Uso de recursos: Utilización de medios físicos para manejar el sistema de software	12 a, 12 b 12 c, 12 d

Fuente: La autora de la investigación