Cuarto Congreso Virtual Iberoamericano de Calidad en Educación a Distancia
EduQ@ 2012

ARQUITECTURA DE SOFTWARE PARA EL PRODUCTO "MIS MEJORES CUENTOS"

Eje 1: La implementación de la EaD en el desafío de la acreditación institucional y los programas de calidad.

Lianet Ballester Jiménez, Universidad de las Ciencias Informáticas, CEDIN, Habana, Cuba. lbjimenez@uci.cu
Adrian García Sánchez, Universidad de las Ciencias Informáticas, FORTES, Habana, Cuba. agciasanchez@uci.cu
Yasmani Ceballos Izquierdo, Universidad de las Ciencias Informáticas, FORTES, Habana, Cuba. yceballos@uci.cu
Lisandra Guibert, Universidad de las Ciencias Informáticas, FORTES, Habana, Cuba. lguilbert@uci.cu

Autor para correspondencia: lbjimenez@uci.cu

Resumen

La arquitectura de software es una disciplina que día a día brinda positivos resultados en el proceso de desarrollo de software, fijando normativas que disminuyen los errores en este delicado proceso. El presente trabajo tiene como objetivo proponer una arquitectura para el producto ¨Mis Mejores Cuentos ¨ que responda a cualidades como la integrabilidad, la portabilidad, la mantenibilidad, entre otras. La arquitectura propuesta proporciona los elementos necesarios para el desarrollo de un producto educativo, sobre plataforma web utilizando

herramientas libres. En el desarrollo se utilizó el patrón arquitectónico MVC en una de sus variantes para la arquitectura del software educativo cubano. Se concibe además la utilización del lenguaje de modelado orientado a la descripción de las vistas del software educativo y se seleccionan las herramientas y tecnologías que serán utilizadas en el desarrollo del producto. **Palabras clave**: arquitectura de software; software educativo; tecnologías.

Abstract

The software architecture is a discipline that provides daily positive results in the software development process, setting standards that reduce errors in this delicate process. This paper aims to propose an architecture for the product ¨Mis Mejores Cuentos¨ responding to qualities such as integrability, portability, maintainability, among others. The proposed architecture provides the necessary elements for developing an educational product on web platform using free tools. During development, architectural pattern MVC is used in one of its variants to the Cuban educational software architecture. It also sees the use of language oriented modeling describing the views of educational software and there are select tools and technologies that will be used in product development.
Keywords: software architecture; educational software; technologies

1. Introducción

“Una arquitectura de software sirve como un puente entre los requerimientos del sistema y su actual implementación” (Garlan, 1994). El objetivo principal de la arquitectura del software es aportar elementos que ayuden a la toma de decisiones y al mismo tiempo, proporcionar conceptos y un lenguaje común que permita la comunicación entre los equipos multidisciplinarios que participan en un proyecto.

Para el año 1992 aproximadamente (Dewayne E, 1994) no era común que los proyectos tuvieran una arquitectura de software (en lo adelante, AS) documentada, pero por la necesidad de la organización del desarrollo de programas informáticos, la reutilización de diseño y códigos, y el mejor entendimiento del software, se hace

necesario la realización de una descripción de la arquitectura al desarrollar un software. Con una incorrecta e inadecuada descripción de la AS un proyecto de desarrollo tiende a fracasar. La Universidad de las Ciencias Informáticas (UCI) está llamada a respaldar las necesidades de producción de software en las diversas ramas de la economía cubana y se abre camino en el desarrollo de productos para otros países que deseen comercializar con esta institución. En este sentido se desea lograr que el trabajo en los proyectos productivos sea satisfactorio, que los productos se obtengan con la calidad requerida y que cumplan con las expectativas de los clientes, favoreciendo que la industria cubana del software se convierta en un renglón fundamental de la economía del país, por lo que una correcta y precisa descripción de la arquitectura de estos productos es de vital importancia para su desarrollo.

Dentro de la infraestructura productiva de la UCI se encuentran los centros de producción especializados en distintos productos y servicios informáticos. Uno de ellos es el Centro de Tecnologías para la Formación (FORTES) que contribuye al desarrollo de aplicaciones educativas siguiendo una línea de desarrollo libre. El proyecto Producción de Recursos Didácticos pertenece a este centro y tiene como propósito crear el producto “Mis Mejores Cuentos” cuyo objetivo es digitalizar los cuentos infantiles clásicos para los niños de 3 a 5 años de edad. Para este producto se realizó una primera versión de la descripción de la AS que no incluyó un correcto diseño de clases pues los miembros del proyecto no estaban capacitados para ello; las librerías utilizadas para facilitar el trabajo con Javascript (Prototype 1.6.0.3 y Script.aculo.us 1.8.2) carecían de numerosas extensiones (plugins), además de que no se planteó con especificidad cómo utilizar el patrón arquitectónico Modelo Vista Controlador (MVC). Al no utilizar una programación orientada a objetos no es posible fomentar la reutilización del código que es uno de los pilares del desarrollo de software con alta calidad. Un diseño de clases adecuado permite agilizar este proceso además que facilita el trabajo en equipo y la creación de sistemas más complejos por lo que esta descripción inicial de AS no se corresponde con los modelos arquitectónicos para plataformas web y los nuevos paradigmas de programación utilizando herramientas libres. En este sentido, el presente artículo tiene como objetivo definir una arquitectura de software para el producto “Mis Mejores Cuentos” estableciendo los elementos necesarios para el desarrollo de un producto educativo sobre plataforma web con herramientas libres.

2. Materiales y métodos o Metodología computacional

Métodos de investigación

Desde el punto de vista metodológico se utilizan los siguientes métodos teóricos: Analítico-Sintético: para realizar un estudio detallado del objeto, logrando resumir y exponer los resultados obtenidos del análisis, así como precisar las características del modelo arquitectónico propuesto. Análisis Histórico-lógico: Se utilizará para estudiar la evolución histórica y desarrollo del objeto de estudio de la investigación.

Estilos arquitectónicos

Un estilo arquitectónico define las reglas generales de la organización en términos de un patrón y las restricciones en la forma y la estructura de un grupo numeroso de sistemas de software, son entidades que ocurren en un nivel sumamente abstracto y puramente arquitectónico.

Patrones arquitectónicos

Los patrones de arquitectura expresan el esquema fundamental de organización para sistemas de software. Proveen un conjunto de subsistemas predefinidos, especifican sus responsabilidades e incluyen reglas y guías para organizar las relaciones entre ellos. Ayudan a especificar la estructura fundamental de una aplicación (Reynoso, 2004).

Modelo - Vista - Controlador (MVC)

El patrón MVC separa el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres partes diferentes:

Modelo: El modelo administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador).

Vista: Maneja la visualización de la información**.**

Controlador: Interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado.

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases. Esta separación permite construir y probar el modelo independientemente de la representación visual.

Modelo Vista Controlador MM (MVCMM)

Desde el surgimiento del MVC y con el paso de los años se han desarrollado diferentes variantes, dentro de las que se encuentra la modificada para Aplicaciones Multimedia, conocida como MVCMM. En esta variante se diversifica la clase modelo incorporando al esquema dos nuevos tipos de clases: la modelo dinámica y la modelo estática, la cual a su vez se subdivide en las clases media y lógica de la aplicación (Ciudad Ricardo, 2007).

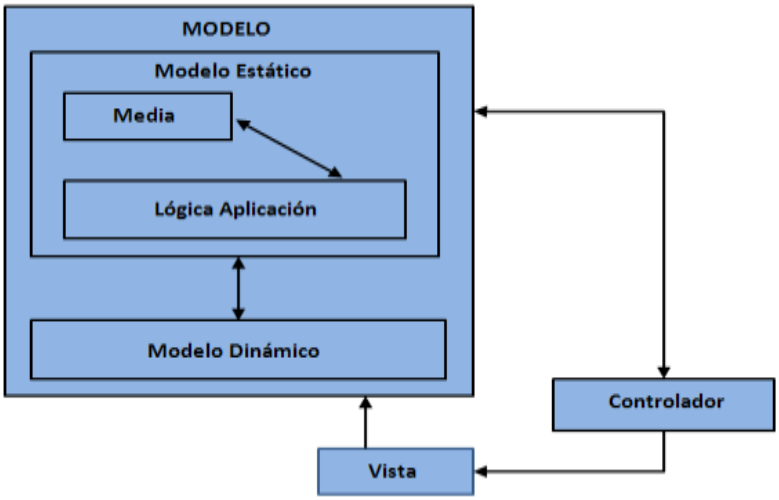

Figura 1. Estructura del MVCMM

Modelo Vista Controlador Entidad (MVC-E)

El MVCMM para un mejor entendimiento y comprensión de aplicaciones educativas recibe un conjunto de transformaciones, dando como resultado, el patrón Modelo – Vista – Controlador – Entidad (MVC-E) como propuesta arquitectónica del software educativo donde se descargan las responsabilidades de la clase modelo concernientes al procesamiento y almacenamiento de la información persistente de las aplicaciones, a una nueva clase incorporada al modelo denominada Modelo Entidad, con dos tipos fundamentales, la clase Modelo – Entidad – Media y Modelo – Entidad - Persistente. La primera de estas con la responsabilidad de agrupar las clases que identifican las medias y su árbol de jerarquía en la aplicación y la segunda tienen como responsabilidad la gestión de la información persistente, que antes sobrecargaba a la clase Modelo del patrón MVC original. Esta variación a su vez sustenta las características actuales de los sistemas multimedia como son: comunicación con bases de datos, archivos XML, archivos JSON, o sistemas externos (Ciudad Ricardo, 2007).

Desde el punto de vista de que la arquitectura de la aplicación estará basada en los principios de la Programación Orientado a Objetos (POO). Los principales elementos de la arquitectura estarán centrados hacia los objetos y serán estos las unidades de modelado a partir de las clases que los definen interactuando a través de funciones y métodos. Por este motivo se selecciona el estilo arquitectónico de Llamada y Retorno y como patrón arquitectónico el Modelo Vista Controlador Entidad como variante de solución para las aplicaciones educativas cubanas.

Herramientas y tecnologías

Para el desarrollo del producto “Mis Mejores Cuentos” se propone utilizar las siguientes herramientas y tecnologías:

Lenguaje de programación web

Para la implementación del sistema se utilizará Javascript como principal lenguaje, por ser libre, por sus características como lenguaje que soporta objetos y por las grandes potencialidades que brinda para manejar los eventos en el entorno web.

Entorno de desarrollo integrado: Aptana

Aptana es un IDE para el desarrollo web de código abierto. Incorpora características completas, sincronización, administración de proyectos, funciones mediante plugins, posibilidad de realizar subidas y bajadas de ficheros a un servidor vía File Transfer Protocol FTP y Secure File Transfer Protocol SFTP, soporte para los sistemas operativos Microsoft Windows, Mac OS X y Linux en sus versiones/distribuciones más recientes (Pérez Chirino, 2010).

El programa brinda diversas utilidades que favorecen el desarrollo de distintas aplicaciones Web, en especial archivos HyperText Markup Language (HTML). Vigila y compila los cambios que se le realicen al proyecto en tiempo real. Permite comprobar la compatibilidad de las funciones con los diferentes navegadores, multiplataforma.

Metodología de desarrollo de software: RUP

Resulta ser la metodología más indicada para guiar el proceso de desarrollo de software pues esta metodología permite generar todos los modelos como principales artefactos en cada uno de los flujos de trabajo, garantizando además una arquitectura robusta desde las primeras etapas del ciclo de vida (Jacobson, 2004).

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software. Además de la amplia documentación que brinda, factor clave para el proyecto pues le aporta requisitos de calidad, escalabilidad y organización al software a desarrollar.

Herramienta CASE: Visual Paradigm

Visual Paradigm es una herramienta de modelado que está diseñada para un gran número de usuarios, incluyendo ingenieros de sistemas, analistas de sistemas, analistas de negocio, arquitectos de sistemas. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite construir todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Una de las características más importantes de Visual Paradigm es que es multiplataforma (Paradigm, 2007).

Lenguaje de modelado: ApEM-L

El Lenguaje para la Modelación de Aplicaciones Educativas y Multimedia (ApEM-L) se presenta como una extensión de UML que permite incorporar a este los elementos fundamentales del proceso productivo UCI, en ApEM-L se incorporan los elementos más significativos de OMMMA-L, para de esta forma lograr una extensión consistente para la modelación de aplicaciones educativas. ApEM-L no modifica la semántica del lenguaje base UML, sino que trabaja en estereotipos restrictivos, por lo que a su vez produce modificaciones descriptivas y decorativas en la representación de los componentes del lenguaje base (Ciudad Ricardo, 2007). Los conceptos y modelos de ApEM-L están agrupados en las siguientes áreas conceptuales: Presentación, Estructura lógica, Comportamiento dinámico y Gestión del modelo. Se selecciona como lenguaje de modelado APEM-L 1.5 porque es un lenguaje orientado a la descripción de las vistas del software educativo, responde además a una mejor modelación de aplicaciones educativas ya que en él se incorporan nuevos diagramas y estereotipos que permiten una mejor comprensión del funcionamiento de aplicaciones con estas características.

Framework de desarrollo

JQuery es una librería programada en Javascript que trabaja con los elementos del DOM, maneja eventos, y crea animaciones sencillas. Además de esto permite ejecutar funciones sin necesidad de recargar la página, manipula las hojas de estilos, proporciona un conjunto de funciones para realizar animaciones con muy pocas líneas de código, se pueden añadir extensiones para aumentar su funcionalidad.

Se utilizará JQuery como framework de desarrollo por su sistema expandible mediante plugins entre los que se encuentran el JQuery UI encargado de la implementación de elementos visuales comúnmente utilizados en aplicaciones web, por la estabilidad característica que acompaña a JQuery en todos los navegadores, por su popularidad entre los desarrolladores web de todo el mundo, y un grado de penetración en el mercado muy amplio. Además es un producto serio, estable, bien documentado y con un gran equipo de desarrolladores a cargo de la mejora y actualización del mismo.

Sistemas de intercambio de datos

Los sistemas gestores de base de datos son un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad y confidencialidad aunque pueden provocar un incremento en el consumo de recursos de hardware y software, lo cual traería problemas para máquinas con pocas prestaciones. Una posible alternativa que se analiza en esta investigación son los formatos de intercambio de datos JSON y XML, los que pueden ser utilizados como un mecanismo ligero de almacenamiento de datos. Sin embargo es imposible que el navegador interactúe con los ficheros JSON y XML sin utilizar la tecnología AJAX.

Teniendo en cuenta las características entre JSON y XML, se decide seleccionar a JSON. Primeramente los datos en formato JSON ocupan mucho menos espacio que en XML. Además cuando se obtiene una respuesta en formato XML se debe recorrer todo el XML, acción por lo general engorrosa y que requiere continuas llamadas recursivas incrementando la utilización de recursos de hardware. JSON representa mejor la estructura de los datos y requiere menos codificación y procesamiento.

3. Resultados y discusión

Una vez definidas las herramientas y tecnologías a utilizar se detallarán los aspectos significativos para la arquitectura de este producto así como la organización de todos los componentes que forman parte de la misma.

Propósito

Ofrecer una visión de la arquitectura del producto “Mis Mejores Cuentos” con el objetivo de representar las principales decisiones arquitectónicas dentro del sistema y explicar detalladamente la estructura que tendrá la aplicación, proporcionando una descripción arquitectónica comprensiva del sistema que servirá de guía (a partir de las diferentes vistas) para posteriores iteraciones.

Los objetivos y restricciones arquitectónicas vienen dados en gran medida por los requisitos no funcionales del sistema, que no son más que las propiedades o cualidades que el producto debe tener.

- Se requiere para el funcionamiento del sistema disponer de un servidor con el sistema operativo Linux y el servidor web Apache 2.2 instalado. Los usuarios del sistema deberán contar con un navegador o browser.
- El sistema será probado, instalado y configurado por los especialistas, que también se ocuparán de su mantenimiento.
- El diseño de la interfaz del sistema deberá ser sencillo, no estará sobrecargado de contenido, pero debe tener elementos visuales que capten la atención y que de forma metafórica representen una idea visual, para lograr un mayor entendimiento a los usuarios finales a los que va dirigido este producto niños entre 3 y 5 años.
- El sistema será implementado con el lenguaje de programación JavaScript haciendo uso de la librería JQuery.
- Las imágenes serán .jpeg o .gif.

La aplicación estará organizada por carpetas según los tipos de archivos. Como se muestra en la siguiente figura:

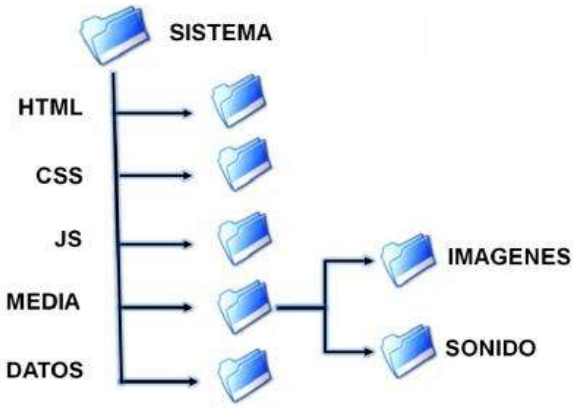

Figura 1. Estructura del sistema

En la siguiente figura se muestra la propuesta de desarrollo a seguir para la modelación de una aplicación educativa aplicando los diagramas de ApEM-L.

Figura 2. Propuesta de desarrollo de ApEM-L

El principio de organización que propone ApEM-L para la construcción de un sistema se basa en la división de la aplicación por subsistemas. Cada subsistema va a estar compuesto por una o más Vistas de Presentación. Las relaciones entre los subsistemas van a ser representados mediante la Vista de Gestión del Modelo (VGM). Para la construcción del producto “Mis Mejores Cuentos” se han identificado 3 subsistemas arquitectónicamente significativos, que se muestran a continuación:

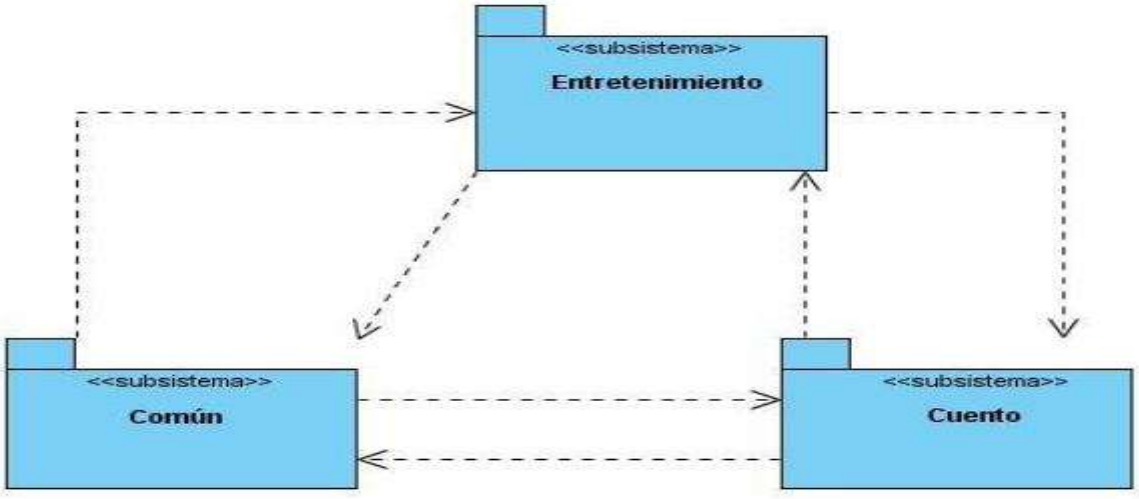

Figura 3. Vista de Gestión del Modelo

Sistema de vistas de ApEM-L

Tanto ApEM-L como UML; no establecen ninguna línea entre los diferentes conceptos y construcciones del lenguaje, pero por conveniencia del primero, este se ha dividido en varias vistas, modelando cada una de estas construcciones que representan un aspecto del sistema. La división ha sido sobre la base de las áreas conceptuales como se puede apreciar en la tabla siguiente.

Área	Vista arquitectónica	Diagramas
Presentación	Vista de Presentación	Diagrama de Estructura de Navegación
		Diagrama de Estructura de Presentación
Estructura lógica	Vista Estática	Diagrama de clases
	Vista de arquitectura	Diagrama de componentes
		Diagrama de despliegue
Comportamiento dinámico	Vista de comportamiento	Diagrama de actividad
		Diagrama de secuencia
		Diagrama de colaboración
		Máquina de estados
Gestión del modelo	Vista de gestión del modelo	Diagrama de clases

Tabla1: Vistas y diagramas de ApEM-L 1.5

Vista de Presentación

La Vista Presentación ha sido incorporada completamente al lenguaje base UML,

para permitir utilizar la semántica original de dicho lenguaje en la construcción de estructuras lógicas de presentación y navegación, incorporando un conjunto de estereotipos restrictivos y descriptivos para una mejor modelación, construyendo el diagrama de estructura de navegación (DEN) y el diagrama de estructura de presentación (DEP).

Vista Estática

La Vista Estática de ApEM-L está compuesta por el diagrama de clases, presentando algunas modificaciones para este tipo de diagrama, que consisten en dividirlo en dos grandes zonas, la de la izquierda dedicada al árbol jerárquico de las clases modelo entidad medias que representan los recursos mediáticos de la aplicación y en la zona de la derecha del diagrama, las clases que controlan la lógica del negocio de la aplicación propiamente dicha. La zona de la derecha se divide a su vez en 4 zonas, la primera dedicada a las clases vista, la contigua a esta y en el extremo superior derecho dedicada a las clases controladoras, inmediatamente debajo de esta sección, la destinada a las clases modelo, quedando una banda inferior derecha dedicada en su extremo derecho a las clases modelo entidad persistentes para el tratamiento de la información persistente de la aplicación; y en el extremo izquierdo las clases correspondientes al Lenguaje de Alto Nivel (HLL).

Vista de arquitectura

La vista de arquitectura está compuesta por el diagrama de componentes y el diagrama de despliegue. El último de los mencionados no sufre cambios en ApEM-L, no así el de componentes donde se incorporan restricciones en los tipos de componentes. Al seguir la arquitectura propuesta por el patrón MVC-E, normalmente los componentes podrán ser organizados por paquetes, que identificarían unidades físicas de encapsulamiento del código (Ciudad Ricardo, 2007).

Validación de la arquitectura

Uno de los factores que determina el éxito o fracaso de un sistema de software, es su arquitectura. El diseñar debidamente una arquitectura de software, garantiza que el sistema de software cumpla con uno o varios atributos de calidad. Si la arquitectura no se diseña de forma apropiada, el sistema de software resultante no logrará sus objetivos. Dicho conocimiento tardío, implica tomar demasiados riesgos innecesarios, un ejemplo es, descubrir fallas en el sistema de software debido a que en la fase de diseño no se eligió apropiadamente una arquitectura. Para reducir tales riesgos, es recomendable llevar acabo evaluaciones a la arquitectura.

Para la evaluación de la arquitectura de software del producto “Mis Mejores Cuentos” se selecciona como método de evaluación Active Reviews for Intermediate Design (ARID), que utiliza la técnica de evaluación basada en

escenarios con el instrumento perfiles, donde se realiza un análisis de los principales escenarios dentro de la arquitectura del sistema. Este método permite realizar evaluaciones a diseños parciales en etapas tempranas del desarrollo, además aprovecha las mejores características de otros métodos, ya que es un híbrido entre Architecture Trade-off Analysis Method (ATAM) y Active Design Review (ADR). Para realizar la evaluación se seleccionaron los atributos de calidad más importantes en aquellos escenarios de acuerdo a un perfil en específico.

Atributos de calidad seleccionados: Desempeño, Mantenibilidad, Integrabilidad, Portabilidad y Disponibilidad. Hacer un adecuado balance de estos atributos entre otros, es esencial para obtener un sistema que cumpla las expectativas de las distintas partes interesadas. A partir de todos los elementos que se han reflejado, se puede afirmar que la solución permite cumplir con los atributos Rendimiento, Mantenibilidad, Integrabilidad, Portabilidad y Disponibilidad. Teniendo en cuenta lo expuesto anteriormente, se define de manera general que el diseño arquitectónico evaluado de manera parcial, dentro de una etapa temprana del desarrollo, resulta aceptable.

Conclusiones

La propuesta presentada constituye la organización estructural y funcional del software educativo “Mis Mejores Cuentos” basado en atributos de calidad logrando un producto multimedia sobre plataforma web con herramientas libres.

La evaluación de la propuesta siguiendo el método ARID permitió demostrar que la arquitectura es aceptable de acuerdo a los indicadores de calidad evaluados y permite aplicarse a productos con características similares.

Referencias

CIUDAD RICARDO, FEBE ÁNGEL. *ApEM – L como una nueva solución a la modelación de aplicaciones educativas multimedias en la UCI.* Ciudad de La Habana : s.n., 2007.

DEWAYNE E, PERRY Y L. WOLF, ALEXANDER. *“Foundations for the study of software architecture”.* s.l. : ACM SIGSOFT, 1994.

GARLAN, DAVID Y SHAW, MARY. *An Introduction to SoftwareArchitecture.* Pittsburgh : Canegie-Mellon University, 1994.

JACOBSON, IVAR, BOOCH, GRADY Y RUMBAUG, JAMES. *El Proceso Unificado de Desarrollo de Software.* La Habana : Félix Varela, 2004.

PARADIGM, VISUAL. Visual Paradigm for UML (ME) - (Paradigma Visual para UML (ME)) (Visual Paradigm for UML (ME)) 6.0. [En línea] 2007. [Consultado el: 12 de noviembre de 2010.]

http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p/.

PÉREZ CHIRINO, YOSBEL. *Herramienta para la gestión de torneos ajedrecísticos, versión 2.0.* La Habana : s.n., 2010.

REYNOSO, CARLOS Y KICILLOF, NICOLÁS.*Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* UNIVERSIDAD DE BUE

CURRÍCULUM

Lianet Ballester Jiménez
Ing. Ciencias Informáticas.
Arquitecta del Proyecto “Producción de recursos didácticos”.