COMPONENTES PERZONALIZABLES PARA EL DESARROLLO DE MULTIMEDIA CON TECNOLOGÍAS LIBRES

CUSTOMIZABLE COMPONENTS OF THE DEVELOPMENT OF MULTIMEDIA WITH FREE TECHNOLOGIES

Daniel Díaz León[1], Susana Becerra Rodriguez[2]

1 Universidad de las Ciencias Informáticas, Cuba, ddleon@uci.cu, Carretera a San Antonio de los Baños, Km 2 1/2. Torrens, La Lisa, La Habana

2 Universidad de las Ciencias Informáticas, Cuba, sbecerra@uci.cu

RESUMEN:

El proyecto Marco de trabajo de desarrollo de multimedia con tecnologías libres posee un conjunto de componentes básicos que le permiten al usuario crear una multimedia. En la actualidad los componentes poseen plantillas de diseño predeterminadas, por lo que no se ajusta a los requerimientos del usuario. Además, poseen baja adaptabilidad en pantalla para los dispositivos móviles. Por esta razón la presente investigación tiene como objetivo desarrollar componentes personalizables para el Marco de trabajo de desarrollo de multimedia con tecnologías libres. Los mismos serán integrados a un entorno de desarrollo para la confección de multimedia. Para guiar el desarrollo de los componentes se utilizó Proceso Unificado Ágil (AUP), generándose los artefactos fundamentales que propone la metodología para cada etapa de trabajo. Se utilizó HTML 5, CSS 3 y JavaScript como lenguajes de programación; como herramientas y tecnologías se emplearon Visual Paradigm 8.0 para el modelado UML, AngularJS 1.4.9, Bootstrap 3.1.1, Nodejs 0.12 y WebStorm 11.0. Como resultado, los usuarios podrán contar con plantillas de diseño cuyos elementos son configurables y se ajusten a sus necesidades, además de facilitar la confección de todo tipo de multimedia. Los componentes personalizables son adaptables a los diferentes dispositivos móviles y pueden ser reutilizados en productos que dentro de su arquitectura tecnológica empleen AngularJS. Se realizaron pruebas a la propuesta de solución para detectar las no conformidades y erradicarlas. Una vez concluidas se pudo apreciar que los componentes desarrollados poseen la calidad y funcionalidad requerida por el cliente.

Palabras Clave: componente personalizable, marco de trabajo, multimedia.

ABSTRACT:

The project Framework of multimedia development work with free technologies has a set of basic components that allow the user to create a multimedia. Currently the components have predetermined design templates, so it does not meet the user's requirements. In addition, they have low screen adaptability for mobile devices. For this reason, the present research aims to develop customizable components for the framework of multimedia development work with free technologies. They will be integrated into a development environment for the production of multimedia. To guide the development of the components Agile Unified Process (AUP) was used, generating the fundamental artifacts proposed by the methodology for each stage of work. HTML 5, CSS 3 and JavaScript were used as programming languages; as tools and technologies, Visual Paradigm 8.0 was used for UML modeling, AngularJS 1.4.9, Bootstrap 3.1.1, Nodejs 0.12 and WebStorm 11.0. As a result, users can count on design templates whose elements are configurable and adapt to their needs, as well as facilitating the preparation of all types of multimedia. The customizable components are adaptable to different mobile devices and can be reused

in products that use AngularJS within their technological architecture. The proposed solution was tested to detect nonconformities and eradicate them. Once concluded it was possible to appreciate that the developed components possess the quality and functionality required by the client.

KeyWords: customizable component, framework, multimedia.

1. INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones (TIC) son utilizadas en distintas esferas de la sociedad como son: salud, deporte, economía, defensa y educación. En la actualidad ha sido necesario vincular la educación con las tecnologías, debido al gran avance tecnológico presente en el mundo. Como consecuencia, a los profesores les ha sido necesario utilizar herramientas informáticas que sirvan de apoyo a las clases, surgiendo así el software educativo.

El software educativo se caracteriza por ser interactivo, debido al empleo de multimedia, fotografías, juegos, diccionarios especializados, ejercicios y cuestionarios con el fin de evaluar a los estudiantes. En la Universidad de las Ciencias Informáticas (UCI) se crean productos informáticos de este tipo, con el propósito de ayudar en el aprendizaje de los estudiantes. En la Universidad existen varios centros de producción, como el Centro de Tecnologías para la Formación (FORTES), perteneciente a la facultad 4. El mismo tiene como misión desarrollar tecnologías que permitan ofrecer servicios y productos. Con el fin de brindar soluciones de formación aplicando las TIC, a todo tipo de instituciones con diferentes modelos de formación y condiciones tecnológicas. Entre los proyectos que pertenecen a FORTES se encuentra el Marco de trabajo de desarrollo de multimedia con tecnologías libres.

El Marco agrupa un conjunto de componentes que facilitan la confección de una multimedia, disminuyen el tiempo de desarrollo, son multiplataforma y adaptables a los dispositivos móviles. También pueden ser reutilizados en los productos que dentro su arquitectura tecnológica emplee AngularJS. En la actualidad estos componentes sólo poseen plantillas de diseño predeterminadas. Esto trae consigo que en ocasiones no se ajusten a los requerimientos del usuario. Así como también posee baja adaptabilidad en pantalla para los diferentes dispositivos. Además, impide que el usuario pueda interactuar con el formato de texto de la aplicación.

Descrita la situación problémica se plantea como problema a resolver: ¿Cómo contribuir al proceso de creación de multimedia dentro del marco de trabajo de desarrollo de multimedia con tecnologías libres?

Se establece como objeto de estudio el proceso de desarrollo de multimedia con tecnologías libres.

Se define como campo de acción de la investigación los componentes personalizables para la creación de multimedia con tecnologías libres.

El objetivo general de la investigación es desarrollar componentes personalizables para el marco de trabajo de desarrollo de multimedia con tecnologías libres.

Los **objetivos específicos** a cumplir son:

- Elaborar el marco teórico de la investigación a partir del estudio del estado del arte sobre el tema.
- Implementar los componentes personalizables para el Marco de trabajo de desarrollo de multimedia.
- Realizar las pruebas necesarias para garantizar la funcionalidad y calidad de los componentes.

Las **tareas a cumplir** son:

- Revisión bibliográfica relacionada con el problema de la investigación.
- Elaboración del diseño teórico de la investigación.
- Diseño de los componentes personalizables a partir del análisis de los requerimientos.
- Implementación de los componentes personalizables.
- Realización de las pruebas con el fin de garantizar la funcionalidad y calidad de los componentes.

La presente investigación está sustentada por la **hipótesis**: Si se incorporan los componentes personalizables al marco de trabajo de desarrollo de multimedia, se contribuirá con el proceso de creación de software multimedia con tecnologías libres.

2. CONTENIDO

2.1 Métodos utilizados

2.1.1 Métodos teóricos

Histórico- Lógico: se empleó en el estudio de la evolución del proceso de desarrollo de multimedia con tecnologías libres.

Analítico- Sintético: se utilizó para realizar el estudio bibliográfico acerca del objeto de estudio de la investigación, con el propósito de definir las características, herramientas y tecnologías de la propuesta

de solución.

2.1.2 Métodos empíricos

Entrevista: se utilizó para identificar las necesidades existentes en la línea de Producción de Recursos Educativos del centro FORTES y de esta manera definir el objetivo que estará dirigido a la propuesta de solución.

Observación: se utilizó para identificar algunas características de la propuesta de solución, con el propósito de realizar un control adecuado de la implementación de los componentes personalizables.

2.2 Conceptos asociados al dominio del problema

Multimedia

El término multimedia hace referencia al uso combinado de diferentes medios de información como texto, imagen, sonido, animación y video. Alguna de las características que presentan las multimedia son [48]:

- Interactividad: es la comunicación recíproca, acción y reacción existente entre la aplicación multimedia y el usuario.
- Ramificación: es la capacidad del sistema multimedia para responder a las preguntas del usuario.
- Transparencia: la tecnología tiene que permitir la utilización de los sistemas multimedia de manera sencilla y rápida.
- Navegación: es la posibilidad que tiene el usuario de desplazarse por la información de forma adecuada sin perderse por la aplicación multimedia.

Medios de información

Los medios de información juegan un papel fundamental dentro de cualquier software, pueden potenciar la memoria visual y auditiva, además facilita el nivel de aprendizaje del usuario. A continuación, se describen los medios de información [48]:

Texto: el texto tiene como función principal favorecer la reflexión y profundización en los temas. Alguna de las ventajas que proporciona su inclusión en las aplicaciones multimedia es desarrollar la comprensión lectora y la fluidez verbal del usuario.

Imágenes estáticas: ilustran y facilitan la información que se desea transmitir en una multimedia.

Imágenes dinámicas, videos o animaciones: transmiten de forma visual secuencias completas de contenido, ilustrando una parte de la información que tiene sentido propio. La animación permite un mayor control de las situaciones, a través de esquemas y figuraciones que la imagen real no refleja en los videos.

Sonido: son incorporados en las multimedia para facilitar la comprensión de la información. Estos pueden ser locuciones orientadas a completar el significado de las imágenes, música y efectos sonoros para conseguir un efecto motivador en el usuario.

Iconográficos: permiten la representación de palabras, conceptos e ideas, mediante imágenes y gráficos.

Gráfico: cualquier elemento que pueda ser presentado en la pantalla de una computadora puede emplearse como un gráfico en un documento multimedia.

Los medios de información pueden ser gestionados por componentes, razón por la cual el desarrollo de multimedia puede ser analizado como un proceso basado en componentes. Un componente es una parte reemplazable, casi independiente y no trivial de un sistema que cumple una función clara en el contexto de una arquitectura bien definida [14]. Entre las principales ventajas que ofrece la utilización de componentes se encuentran la reutilización en diversas aplicaciones, reducción de costos y el mejoramiento de la calidad.

Personalización

La personalización de aplicaciones es entendida como la capacidad de alteración dinámica con el fin de proporcionar al usuario la impresión de estar trabajando con una aplicación específicamente diseñada para dar satisfacción a sus necesidades particulares. [45]

Componente personalizable

Definido los términos componente y personalización se procede a definir componente personalizable. Debido a que no existe una definición de componente personalizable los autores de la presente investigación lo definen de la siguiente forma:

Un componente personalizable es una parte reemplazable del software, le brinda al usuario la posibilidad de ajustar la interfaz acorde a sus necesidades y preferencias. Además, adapta su estructura y contenido al dispositivo de acceso.

Marco de trabajo

El término marco de trabajo o *framework*, hace referencia a una estructura de software compuesta por componentes personalizables e intercambiables para el desarrollo de una aplicación. Es decir, un marco de trabajo se puede considerar como una aplicación genérica incompleta y configurable a la que se puede añadir las últimas piezas para construir una aplicación concreta. [49]

Los principales objetivos de un marco de trabajo son:

- Disminuir el tiempo del proceso de desarrollo.
- Reutilizar código ya existente.
- Promover buenas prácticas de desarrollo como el uso de patrones.

2.3 Metodología, tecnología y herramienta

2.3.1 Metodología de desarrollo de software

Proceso Unificado Ágil (AUP)

Constituye una versión simplificada del Proceso Unificado Racional (*Rational Unified Process*, RUP), desarrollada por Scott Ambler. Esta metodología combina procesos propios del concepto unificado tradicional con técnicas ágiles, con el objetivo de mejorar la productividad. Permitiendo así describir de manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio. AUP aplica técnicas ágiles [9]:

Al igual que RUP, en AUP se definen cuatro fases que se desarrollan de manera consecutiva:

a. Inicio.
b. Elaboración.
c. Construcción.
d. Transición.

AUP define 7 disciplinas, de ellas cuatro son dirigidos para la ingeniería y 3 para la gestión de proyectos. La Universidad realiza una variación definiendo nuevas disciplinas.

Las disciplinas propuestas por AUP: gestión de configuración, gestión de proyectos y entorno, se cubrirán de acuerdo a las áreas de procesos que define CMMI-DEV en su versión 1.3. Estas quedan de la siguiente forma: gestión de la configuración, la planeación del proyecto, el monitoreo y control de proyecto.

AUP propone 9 roles en el desarrollo de un proyecto. Pero debido a la adaptación realizada por la Universidad, decide para el ciclo de vida de los proyectos tener 11 roles, de los cuales se mantienen algunos de los propuestos por AUP y unificando o agregando otros para una mejor organización del equipo de desarrollo. A continuación, se muestran los roles resultantes:

a. Jefe de proyecto.
b. Planificador.
c. Analista.
d. Arquitecto de información.
e. Desarrollador.
f. Administrador de la configuración.
g. *Stakeholder* (Cliente/Proveedor de requisitos).
h. Administrador de calidad.
i. Probador.
j. Arquitecto de software.
k. Administrador de base de datos.

2.3.2 Tecnologías y herramientas

Lenguajes de desarrollo

Los lenguajes de programación son un conjunto de reglas, normas, símbolos y caracteres que le permite a un programador la creación de programas y resolver los mismos de manera eficaz [32]. Para el desarrollo de la presente investigación se utilizaron los lenguajes del lado del cliente y del servidor.

Lenguajes del lado del cliente: son aquellos que el programa reside junto a la página web en el servidor, pero es transferido al cliente para que este los ejecute. Es decir, el servidor no interviene en el proceso de crear la página web solicitada por el usuario.

Lenguajes del lado del servidor: los lenguajes son ejecutados por el servidor y lo que se envía al cliente es la respuesta o el resultado de dicha ejecución.

A continuación, se describen los lenguajes que se utilizaron:

HTML (*HyperText Markup Language*): es el elemento de construcción más básico de una página web, se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad. HTML le añade "marcado" a un texto estándar en español. "Hipertexto" se refiere a enlaces que conectan una página web con otra. Con la ayuda de HTML se pueden hacer sitios estáticos y dinámicos. [39]

CSS (*Cascading Style Sheets*): es un lenguaje usado para definir la presentación semántica de un documento estructurado escrito en HTML o XHTML. Se crea con el principal objetivo de separar el contenido de la forma en que se visualiza la página web, permitiéndoles a los diseñadores un mayor control sobre las apariencias de sus páginas. [17]

JavaScript: es un lenguaje de programación interpretado por el navegador web, es compatible con la mayoría de los navegadores modernos (Chrome, Firefox, Internet Explorer, Safari y Opera). Se utiliza para crear páginas web dinámicas, desarrollado por Netscape. Es un lenguaje de script multiplataforma, sencillo, de rápida ejecución, consume poca memoria, fácil de integrar a las páginas web, tiene gran cantidad de efectos visuales. JavaScript permite añadir gran cantidad de efectos visuales y utiliza el DOM (*Document Object Model)* para modificar el HTML con mayor facilidad sin tener que recargar todo el documento. [8]

Para la implantación de la propuesta de solución se utilizó el lenguaje JavaScript del lado del cliente y del servidor.

Generador de proyecto

Un generador es un complemento que se puede ejecutar mediante un comando.

Yeoman v1.6.0: permite generar proyectos web con un solo comando, estos pueden ser de todo tipo Ruby, PHP, JavaScript, etc. Con el propósito de crear proyectos con el *framework* AngularJS se creó un "generador". Un generador es un complemento que se puede ejecutar con el comando "yo". A través del mismo se promueve el "flujo de trabajo Yeoman". Este flujo de trabajo comprende tres tipos de herramientas para mejorar la productividad y satisfacción en la construcción de una aplicación web. [1]

Estas herramientas son:

- La herramienta de andamios (yo[1]).
- La herramienta de construcción (Grunt, Gulp).
- El gestor de paquetes (como Bower).

Bower v1.7.7: es un módulo de Nodejs que puede administrar los componentes que contienen HTML, CSS, JavaScript, fuentes e incluso archivos de imagen. Es un sirviente, con solo indicarle la biblioteca o *framework* que se necesita, él se encarga de descargar las versiones correctas de los paquetes y sus dependencias. Tiene además opciones de búsqueda, actualización y eliminación de assets [46]

Un ejemplo de lo descrito anteriormente es: si se comienza un nuevo proyecto y se va a utilizar jQuery, AngularJS, Bootstrap, Angular-Bootstrap y MomentJS, solo hay que escribir la siguiente instrucción: "bower install jquery angular bootstrap angular-bootstrap momentjs".

Este comando realiza 3 opciones:

- Buscar en su base de datos online la existencia de estos assets.
- Descargarlos hacia la cache (.cache/bower en Linux, %appdata%/Local/bower en Windows) en caso que no se haya descargado.
- Copiarlos hacia el proyecto.

Grunt v0.1.13: es un módulo de Nodejs que permite configurar tareas automáticas y así ahorrar tiempo en el desarrollo y despliegue de aplicaciones web. Alguna de las tareas más comunes a automatizar son [33]:

- Concatenación de archivos (CSS, JS).
- Minificación de archivos (CSS, JS).
- Optimización de imágenes.
- Compilación (SASS -> CSS y otros).
- Pruebas unitarias.

Una de las ventajas que posee Grunt es que es JavaScript al igual que los complementos y utiliza Nodejs para su ejecución, lo que le permite ser multiplataforma.

NPM v2.5.1: es un gestor de paquetes para JavaScript que le permite a los desarrolladores compartir y reutilizar paquetes de código. Además, permite que la actualización del código a compartir sea fácil [19].

Marcos de trabajo

AngularJS v1.4.9: es un *framework* JavaScript de código abierto desarrollado y mantenido por Google. Implementa el patrón de diseño Modelo Vista Controlador (MVC), para crear aplicaciones de una sola página (single-page-applications). AngularJS extiende el tradicional HTML con etiquetas (directivas) propias. Posee un sistema de *databinding* extensible haciendo uso de las directivas y los filtros, lo que propicia que la capa de presentación el código sea más simple y organizado. AngularJS pretende que los programadores que lo usen generen un HTML entendible para aquellas personas que no tengan mucho conocimiento informático y que una vez lo lean sepan que es lo que hace y para qué sirve la aplicación creada [51].

Características de AngularJS:

- Fácil comprensión para los que comienzan a usarlo pues ofrece características sofisticadas para desarrolladores con necesidades complejas.
- El código de aplicaciones creadas con AngularJS siempre está organizado por Modelos, Vistas, Controladores y opcionalmente Servicios.

Ventajas de AngularJS:

- Potente sistema de *templating* incluido en el mismo *framework*.
- Sincronización entre vistas y modelos para crear páginas one-page.
- Uso de directivas para la creación de nuevos atributos o nuevas etiquetas HTML.
- Uso de filtros para alterar la presentación de datos.
- Uso de Servicios que se encargan de la comunicación con el servidor para la consulta de datos.

Jasmine v2.4.1: es un marco de desarrollo impulsado por el comportamiento de JavaScript para probar aplicaciones de AngularJS. No depende de ningún otro *framework* de JavaScript, así como tampoco requiere un DOM. Jasmine posee una sintaxis limpia por lo que se puede escribir fácilmente pruebas [29].

Express v4.13.4: es un *framework* de aplicaciones web Node.js mínima y flexible, que proporciona un conjunto solido de características para las aplicaciones web y móviles. Express define una tabla de enrutamiento que se utiliza para llevar a cabo una acción diferente, basada en el método HTTP y URL.

[1] yo: es un comando de Yeoman.

Además, permite representar dinámicamente páginas HTML [41].

Bootstrap v3.1.1: es un *framework* creado inicialmente por *Twitter* y posteriormente liberado bajo la licencia MIT, para el desarrollo de interfaces web. Incluye HTML y CSS basado en el diseño de plantillas para la tipografía, formularios, botones, tablas, módulos, carruseles de imágenes y muchas otras. Es una herramienta compatible con todos los navegadores modernos (Chrome, Firefox, Internet Explorer, Safari, and Opera). Es adaptable al tamaño de cualquier dispositivo donde se visualice, como los dispositivos móviles, tablets y ordenadores de sobremesa. [47]

Esta versión consta de varias características importantes como:

- Tiene un soporte casi completo con HTML5 y CSS3, flexibilizando la creación de aplicaciones web.
- Permite utilizar un GRID de 12 columnas donde colocar el contenido, posibilitando crear diseños web *responsive* de una forma más fácil.
- Permite la inserción de imágenes responsive con solamente ponerle la clase "img-responsive" a las imágenes, estas se adaptan al tamaño deseado.

Entorno integrado de desarrollo

Un Entorno Integrado de Desarrollo (IDE por sus siglas en inglés), es un entorno de programación que ha sido empaquetado como un programa de aplicación. Es decir, un IDE se compone de un editor de código de programación, un compilador, un intérprete, un depurador y un constructor de interfaz gráfica. [11]

A continuación, se menciona el IDE a utilizar para el desarrollo de la propuesta de solución:

WebStorm v11.0: es un IDE de JavaScript multiplataforma desarrollado por JetBrains[2]. Utilizado para desarrollar aplicaciones web con soporte para las tecnologías JavaScript, Nodejs, HTML y CSS. Posee finalización de código inteligente, detección de errores en el mismo momento en que se escribe. WebStorm tiene soporte para el desarrollo web con *framework* como AngularJS, React, Meteor, entre otros. Algunas de las características principales son [3]:

- Producir código de alta calidad de manera más eficiente, gracias a la finalización de código inteligente, detección de errores y la navegación de gran alcance sobre la marcha.
- Funciona en Windows, Mac OS o Linux con una única clave de licencia.
- Es un software con todas las funciones para construir sitios web que usan herramienta como el depurador, VCS, en terminales y otras herramientas.
- Ofrece soporte para JavaScript, Node.js, ECMAScript 6, mecanografiado, CoffeeScript, Menos, Sass, Stylus y Dardo.
- Combina los mejores resultados de la finalización de código para todos los métodos, funciones, módulos, variables y clases definidas.

Intérprete de JavaScript

Un motor JavaScript (también conocido como intérprete de JavaScript) es la parte del navegador que interpreta y ejecuta el código escrito en el lenguaje de programación JavaScript. [28]

Nodejs v0.12.0: es un intérprete de JavaScript, construido sobre el motor JavaScript v8 de Chrome. Posee un entorno de ejecución multiplataforma de código abierto para el desarrollo del lado del servidor y las aplicaciones de red escalables, contiene un entorno de programación dirigido por eventos, basado en el lenguaje de programación ECMAScript. Ventajas de Nodejs [20]:

- Está basado en eventos, así que toda la filosofía asíncrona utilizada con AJAX en el cliente se puede pasar al servidor.
- Permite utilizar el mismo lenguaje (JavaScript) tanto en el cliente como en el servidor.
- El desarrollo es más rápido.
- La ejecución de test de unidad se puede hacer más rápido.

Visual Paradigm v8.0: es una herramienta CASE, que soporta el modelado mediante UML. Proporciona asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. [18]

2.4 Descripción de la arquitectura

La arquitectura de software consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software. [37]

La arquitectura de la propuesta de solución está determinada por el uso de la tecnología AngularJS. Permitiendo el desarrollo de aplicaciones web siguiendo la arquitectura Modelo Vista Controlador (MVC).

El Modelo Vista Controlador es un patrón arquitectónico de software para el desarrollo de aplicaciones web. Este aísla la lógica de la aplicación de la capa

[2] JetBrains: es una empresa de desarrollo de software cuyas herramientas que están dirigidas a los desarrolladores de software y administradores de proyectos.

de interfaz de usuario. El controlador recibe las peticiones de la aplicación y luego trabaja con el modelo para elaborar los datos que necesita la vista. La vista utiliza los datos preparados por el controlador para generar una respuesta final. [4]

El MVC se puede representar gráficamente como:

- El modelo: es responsable de la gestión de datos de la aplicación. Responde a la solicitud de la vista y con las instrucciones del controlador de actualización de sí mismo.
- La vista: es una presentación de los datos en un formato particular, desencadenada por la decisión del controlador para presentar los datos.
- El controlador: responde a la entrada del usuario y realiza interacciones sobre los objetos del modelo de datos. Este recibe la entrada, valida y a continuación realiza las operaciones de negocios que modifican el estado del modelo de datos.

Algunas de las ventajas del MVC son [15]:

- La separación clara entre los componentes de un programa, permitiendo su implantación por separado.
- La Interfaz de Programación de Aplicaciones (API) muy bien definida. Lo cual permite que cualquiera que utilice el API, podrá reemplazar el Modelo, la Vista o el Controlador sin ninguna dificultad.
- Conexión entre el Modelo y sus Vistas dinámicas; se produce en tiempo de ejecución y no en tiempo de compilación.

2.5 Pruebas del software

Las pruebas de software son un elemento fundamental para la garantía de calidad del software. Además, representan una revisión final de las especificaciones, del diseño y de la codificación. [14]

2.5.1 Pruebas unitarias

En el Generador de proyectos de AngularJS se utiliza Karma para configurar un proyecto de prueba preparada con todas las funcionalidades. Debido al empleo de la tecnología AngularJS en la presente investigación se decide utilizar la herramienta Karma.

El objetivo principal de **Karma** es ofrecer un entorno de pruebas productivas para los desarrolladores. Con el entorno no tienen que establecer las configuraciones, este ofrece un espacio donde los desarrolladores pueden simplemente escribir el código y obtener información inmediata de sus pruebas. [25]

En la figura 1 se muestra las pruebas unitarias realizadas al componente galería de imágenes.

```
'use strict';
describe('Controller: FwGaleriaDeImagenesCtrl', function () {
  beforeEach(module('ndmMeanApp'));
  var FwGaleriaDeImagenesCtrl, scope;
  beforeEach(inject(function ($controller, $rootScope) {
    scope = $rootScope.$new();
    FwGaleriaDeImagenesCtrl = $controller('FwGaleriaDeImagenesCtrl', {
      $scope: scope
    });
    scope.imagenes = [
      {
        'titulo': "Amor",
        'descripcion': 'sentimiento de afecto universal que se tiene hacia una persona, animal o cosa',
        'dirImg': 'Mi Primera Multimedia/componentes/galeria_Img/datos/amor.jpg'
      },
      {
        'titulo': "Amigos",
        'descripcion': 'persona con quien se mantiene una amistad. Una amistad es una relación afectiva entre dos personas, construida
        'dirImg': 'Mi Primera Multimedia/componentes/galeria_Img/datos/amistad.jpg'
      },
      {
        'titulo': "Felicidad",
        'descripcion': 'estado emocional de una persona feliz; es la sensación de bienestar y realización que experimentamos cuando al
        'dirImg': 'Mi Primera Multimedia/componentes/galeria_Img/datos/feliz.jpg'
      }
    ];
  }));
  it('Cantidad de imagenes', function () {
    expect(scope.imagenes.length).toEqual(3);
  });
  it('Generar galeria de imagenes', function () {
    expect(scope.aplicarIMG).toBeDefined();
  });
  it('Comprobar expresion regular en el nombre de la galeria', function () {
    expect("Galeria de Mi multimedia").toMatch('^[a-zA-Z]+(\s*[a-zA-Z]*)*[a-zA-Z]+$');
  });
});
```

Figura 1 Pruebas unitarias realizadas al componente galería de imágenes

Las pruebas unitarias fueron realizadas a todos los componentes personalizables, los cuales conforman la propuesta de solución. Estas fueron ejecutadas satisfactoriamente obteniendo resultados favorables. En la Figura 2 se muestran los resultados obtenidos en las pruebas unitarias de los componentes.

```
Controller: FwAnimacionCtrl
  ✓ Método principal para animar
  ✓ Variable de animaciones no null
  ✓ Variable de estilos no null

Controller: FwGaleriaDeImagenesCtrl
  ✓ Cantidad de imagenes
  ✓ Generar galeria de imagenes
  ✓ Comprobar expresion regular en el nombre de la galeria

Controller: FwGaleriaDeVideosCtrl
  ✓ Cantidad de videos
  ✓ Generar galeria videos
  ✓ Comprobar expresion regular del titulo la galeria

Controller: FwGlosarioDeTerminosCtrl
  ✓ Adicionar letra
  ✓ Guardar palabra y descripcion y validar los mismos
  ✓ Datos no nulos
  ✓ Validar expresiones regulares

Controller: FwMapaCtrl
  ✓ Adicionar y actualizar punto
  ✓ Datos no nulos
  ✓ Cantidad de datos

Controller: FwMenuCtrl
  ✓ Guardar datos
  ✓ Eliminar datos
  ✓ Guardar enlace
  ✓ Guardar desplegable

 Controller: FwTemaCtrl
   ✓ Comprobar sistema Gird
   ✓ Verificar si scope.tomarColumnas esta definido

Chromium 48.0.2564 (Ubuntu 0.0.0): Executed 22 of 22 SUCCESS (0.227 secs / 0.178 secs)
TOTAL: 22 SUCCESS
```

Figura 2 Resultados de las pruebas unitarias de los componentes.

2.5.2 Pruebas de integración

En la presente investigación se realizaron las pruebas de integración a los componentes personalizables. En las pruebas se obtuvo como resultado para una primera iteración una no conformidad, la cual fue resuelta. En una segunda iteración se obtuvo resultados favorables, integrándose los componentes: Galería de imágenes, Galería de videos, Glosario de términos, Tema, Mapa y Menú, con el entorno de desarrollo. A continuación, se muestra la imagen evidenciándose la integración

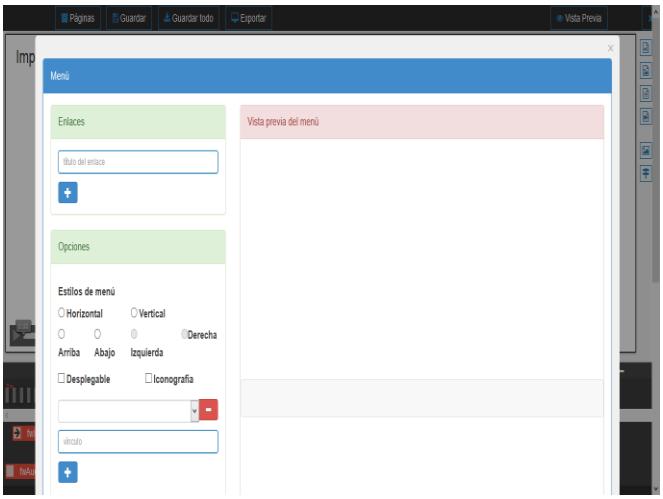


Figura 3 Prueba de integración del componente menú.

2.5.3 Pruebas de aceptación

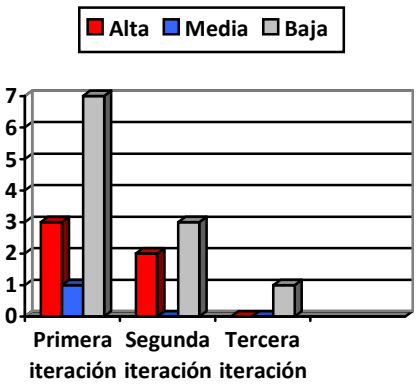
Las pruebas de aceptación fueron realizadas a todos los componentes personalizables que conforman la propuesta de solución. Para ello se efectuaron tres iteraciones, en las cuales se detectaron 17 no conformidades; de estas 5 se clasificaron de complejidad alta, 1 de complejidad media y 11 de complejidad baja.

Las no conformidades fueron clasificadas según su complejidad por lo que se definen de la siguiente forma:

- Alta son los errores en el código o errores de funcionalidad.
- Media son los errores de ortografía o de validación.
- Baja son los errores de interfaz.

A continuación, se expone un gráfico que muestra la relación entre las no conformidades detectadas de complejidad alta, media o baja.

Tabla 1 No conformidades



Una vez concluida cada iteración de pruebas se analizaron por parte del equipo de desarrollo las no conformidades encontradas y se determinaron las que constituían fallas del sistema o prestaban variaciones en las Historias de usuario.

2.6 Resultados

Previo a la implementación de la solución propuesta fueron definidos un total de 40 requisitos funcionales del sistema y 8 requisitos no funcionales. Todos estos requisitos fueron extraídos partiendo de los resultados del estudio previo del estado actual de las soluciones similares existentes y a entrevistas realizadas a los especialistas del centro FORTES, centro encargado del desarrollo de multimedia.

Los componentes personalizables galería de imágenes, galería de videos, glosario de términos, tema, menú, mapa y paquete de animaciones les permiten a los usuarios contar con plantillas de diseño cuyos elementos sean configurables. Por lo que se obtuvo un producto que se ajuste a sus requerimientos, pueden ser ejecutados en cualquier sistema operativo y en diferentes dispositivos. También pueden ser reutilizados en productos que dentro de su arquitectura tecnológica empleen AngularJS. Además, se pueden crear los componentes para la multimedia mediante clic sin tener mucho conocimiento de programación.

3. CONCLUSIONES

El desarrollo de los componentes personalizables en conjunto con el análisis de los resultados obtenidos, posibilitaron obtener las siguientes conclusiones:

- El estudio del estado del arte permitió identificar los conceptos relacionados con el objeto de estudio.
- El análisis de las soluciones similares posibilitó determinar las funcionalidades bases para el desarrollo de los componentes.
- Los artefactos ingenieriles generados permitieron describir la propuesta de solución.
- Para darle cumplimiento al objetivo general, se desarrollaron siete componentes personalizables utilizando las tecnologías y herramientas: AngularJS, Bootstrap, Nodejs, WebStorm, Yeoman, Bower, Grunt, Jasmine, Express y NPM.
- Los componentes personalizables desarrollados permitieron mejorar el proceso de creación de multimedia en el centro FORTES.
- Se aplicaron pruebas unitarias, de integración y aceptación las cuales permitieron determinar y erradicar las no conformidades encontradas.

4. REFERENCIAS BIBLIOGRÁFICAS

1. **Yeoman.** Yeoman. [En línea] 2016. [Citado el:

17 de mayo de 2016.] http://yeoman.io/.

2. **Yenisleidy Fernández Romero, Yanette Díaz González.** Revista Telem@tica. *Revista Telem@tica.* [En línea] enero-abril de 2012. [Citado el: 25 de febrero de 2016.] http://revistatelematica.cujae.edu.cu/index.php/tele/article/viewFile/15/10. ISSN 1729-3804.

3. **WebStorm.** JetBrains. *WebStorm.* [En línea] 2016. [Citado el: febrero de 8 de 2016.] https://www.jetbrains.com/webstorm/features/ide-features.html.

4. **Tutorialspoint.** Angularjs- MVC Architecture. [En línea] 2016. [Citado el: 16 de marzo de 2016.] http://www.tutorialspoint.com/angularjs/angularjs_mvc_architecture.htm.

5. **Thomas Müller, Armin Beer, Martin Klonk, Rahul Verma.** *Probador Certificado. Programa de estudio de nivel básico.* Ciudad de la Habana : s.n., 2010.

6. **Sommerville, Ian.** *Ingeniería de Software. 8va Edición.* 2007. ISBN 10: 0-321-31379-8.

7. —. *Ingeniería de software, 7ma edición.* Madrid : Pearson Educación, S.A., 2005. ISBN: 84-7829-074-5.

8. **Scholz, Florian.** Mozilla Developer Network (MDN). [En línea] 26 de abril de 2015. [Citado el: 1 de marzo de 2016.] https://developer.mozilla.org/es/docs/Web/JavaScript/Acerca_de_JavaScript.

9. **Sánchez, Tamara Rodríguez.** *Metodología de desarrollo para la actividad productiva de la UCI.* La Habana : s.n., 2014.

10. **Rodríguez, Msc. Dayra Iris Hechavarría.** Gestión de Requisitos. *Gestión de Requisitos.* [En línea] 2012. [Citado el: 2016 de febrero de 25.] http://www.monografias.com/trabajos92/gestion-requisitos/gestion-requisitos.shtml#queesunrea.

11. **Ricardo Barrera Urieta, María Elizabeth García Mayo, Joaquín Herrera Nova, Adilene Ríos Urbina.** *Plataformas de Desarrollo de Aplicaciones Móviles.* Costa Grande : s.n., 2014.

12. **Reyes, Rebeca Abigail González, Eugenio, Aarón Augusto Pérez, González, Dulce Fabiola Ramos, Chávez, Mayte Rivera.** *Estereotipos e Interfaces.* Tijuana : s.n., 2014.

13. **Real, Víctor Perlacia y Carballo, Yeidy Martínez.** *Componentes básicos para el marco de trabajo de desarrollo de multimedia con tecnologías libres.* La Habana : s.n., 2015.

14. **Pressman, Roger S.** *Ingeniería de software. Un enfoque práctico. 5ta Edición.* New York : s.n., 2005. ISBN 978-0-07-337597-7.

15. **Portillo, M. del Pilar del Saz.** *Tutorial Patrón MVC.* 2014.

16. **PMBOK.** *Guía de los fundamentos para la dirección de proyectos (Guía del PMBOK). Cuarta Edición.* 2008. ISBN: 978-1-933890-72-2.

17. **Pérez, Javier Eguíluz.** *Introducción a CSS.* 2009.

18. **Paradigm, Visual.** Visual Paradigm. [En línea] 2015. [Citado el: 13 de Enero de 2016.] http://www.visual-paradigm.com/aboutus/newsreleases/vpuml80.jsp.

19. **NPM.** NPM. [En línea] 13 de marzo de 2016. [Citado el: 9 de junio de 2016.] https://docs.npmjs.com/getting-started/what-is-npm.

20. **Nodejs.** Nodejs. [En línea] 6 de febrero de 2015. [Citado el: 5 de febrero de 2016.] https://nodejs.org/en/blog/release/v0.12.0/.

21. **Microsoft.** Microsoft. [En línea] 2015. [Citado el: 5 de abril de 2016.] https://msdn.microsoft.com/es-es/library/dd409390.aspx.

22. **Lennon, Joe.** Cree sitios Web modernos usando HTML5 y CSS3. *Cree sitios Web modernos usando HTML5 y CSS3.* [En línea] 14 de junio de 2011. [Citado el: 4 de febrero de 2016.] https://www.ibm.com/developerworks/ssa/web/tutorials/wa-html5/#ibm-pcon.

23. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : s.n., 1999. ISBN 970-17-0261-1.

24. **Labrada, Sonia Morejón.** Cuadernos de educación y desarrollo. *El software educativo un medio de enseñanza eficiente.* [En línea] julio de 2011. [Citado el: 26 de noviembre de 2015.] http://www.eumed.net/rev/ced/29/sml.htm.

25. **Karma.** Karma. [En línea] 2016. [Citado el: 27 de abril de 2016.] http://karma-runner.github.io/0.8/index.html.

26. **Jorge R. Aguilar Cisneros, Carlos Alberto Fernández-y-Fernández.** *Especificación de Requerimientos para el Desarrollo de Software Automotriz en México.* México : s.n., 2015. ISSN 2314-2642.

27. **Jonathan Nieto, Marco Martínez.** *Sistema Wed Ayni. Estandares de programación. .* 2011.

28. **JavaScript Engines, o motores JavaScript.** JavaScript Engines, o motores JavaScript. [En línea] 2015. [Citado el: 4 de mayo de 2016.] https://moodle2015-16.ua.es/moodle/pluginfile.php/26075/mod_resource/content/5/page_08.htm.

29. **Jasmine.** Jasmine. [En línea] 2016. [Citado el: 9 de junio de 2016.] http://jasmine.github.io/2.4/introduction.html.

30. **Ivar Jacobson, Grady Booch, James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* s.l. : Addison Wesley, 2000. ISBN: 74-7829-036-2.

31. **Informáticas, Portal de la Universidad de las Ciencias.** Portal de la Universidad de las Ciencias Informáticas. [En línea] 2012. [Citado el: 26 de noviembre de 2015.] http://www.uci.cu/?q=mision.

32. **Hernández, M.C. Norma Ramírez, López, M.S.I Graciela Lara y Chávez, M.A.S.I. Salomón Eduardo Ibarra.** *Análisis, diseño y programación de sistemas.* 2014. ISBN 970764803-1.

33. **Grunt.** Grunt. [En línea] 2016. [Citado el: 17 de mayo de 2016.] http://gruntjs.com/getting-started.

34. **González, Rolando Alfredo Hernández León y Sayda Coello.** *El proceso de investigación científica.* Ciudad de La Habana : Editorial Universitaria del Ministerio de Educación Superior, 2011. ISBN 978-959-16.

35. **GitHud.** GitHud. *Principios para escribir JavaScript consistente e idiomático.* [En línea] 2016. [Citado el: 28 de abril de 2016.] https://github.com/rwaldron/idiomatic.js/tree/master/translations/es_ES.

36. **Gelderen, María Marta van.** *Multimedia educativa.* Ciudad Autónoma de Buenos Aires : s.n., 2011.

37. **Fuentes, Inés Meriño.** *Arquitectura de software.* Universidad del Magdalena : s.n., 2012.

38. **Flores, Araceli Soledad Domínguez.** Unidad 1: UML y el proceso unificado. *Desarrollo e implementación de Sistemas de Información.* 2013.

39. **Florentino, Marvin.** Mozilla Developer Network (MDN). [En línea] 24 de febrero de 2016. [Citado el: 1 de marzo de 2016.] https://developer.mozilla.org/es/docs/Web/HTML.

40. **Expto. Oscar Zalazar, Expto. Pedro L. Alfonzo, Lic. Yanina Medina, Osvaldo P. Quintana, Lic. Lucía Salazar.** *Taller de programación 1.* 2012.

41. **Express.** Express. [En línea] 2016. [Citado el: 9 de junio de 2016.] http://expressjs.com/es/.

42. **Datos, Grupo de Ingeniería del Software y Bases de.** *Documentación de requisitos de cliente: Historias de usuario.* Sevilla : s.n., 2013.

43. **CUEVAS, NIELS HENRYK ARANDA.** [En línea] 2014 de junio de 2014. [Citado el: 7 de febrero de 2016.] http://es.slideshare.net/GiancarloAguilarChe/componentes-y-libreras-tpicos-avanzados-de-programacin.

44. **CSS3.** CSS3. [En línea] 25 de enero de 2016. [Citado el: 4 de febrero de 2016.] https://developer.mozilla.org/es/docs/Web/CSS/CSS3.

45. **Cachero, Cristina.** *Personalización de Aplicaciones en OO-H .* 2010.

46. **Bower.** Bower. [En línea] 2012. [Citado el: 17 de mayo de 2016.] http://bower.io/.

47. **Bootstrap.** Bootstrap. [En línea] 2016. [Citado el: 4 de febrero de 2016.] http://getbootstrap.com/.

48. **Belloch, Consuelo.** APLICACIONES MULTIMEDIA. [En línea] 2012. [Citado el: 2 de diciembre de 2015.] http://www.uv.es/bellochc/logopedia/NRTLogo4.pdf.

49. **Aponte, Ángela María Valbuena.** *Guía comparativa de frameworks para los lenguajes HTML 5, CSS y Javascript para el desarrollo de aplicaciones web.* Universidad Tecnológica de Pereira : s.n., 2012. Tesis.

50. **Angel.** Geed the pianet. [En línea] mayo de 2011. [Citado el: 1 de marzo de 2016.] http://geektheplanet.net/5462/patrones-gof.xhtml.

51. **Alvarez, Alberto Basalo y Miguel Angel.** Qué es AngularJS. [En línea] 28 de agosto de 2014. [Citado el: 15 de enero de 2016.] http://www.desarrolloweb.com/articulos/que-es-angularjs-descripcion-framework-javascript-conceptos.html.