Universidad de las Ciencias Informáticas

FACULTAD 6

Sistema de Gestión de Información para la Muestra Joven ICAIC

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Autor: Yohan González Espinosa

Tutores: Ing. Dianela Borrego León

Ing. Leanet Arza Pérez

La Habana, febrero de 2015

"Año 57 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, y le concedo los derechos patrimoniales, con carácter exclusivo.

Para que así conste firmo a los ___días del mes de _____ del año_____.

_____	_____
Firma del Autor	Firma del Tutor
Yohan González Espinosa	Ing. Dianela Borrego León

_____

Firma del Tutor

Ing. Leanet Arza Pérez

DATOS DE CONTACTO

Tutor: Ing. Dianela Borrego León

Graduado en la Universidad de Ciencias Informáticas (UCI) como Ingeniero en Ciencias Informáticas, 2012

Trabajador de la Dirección para la Comercialización y los Negocios
Correo electrónico: dborrego@uci.cu
Teléfono: 837-3865

Tutor: Ing. Leanet Arza Pérez

Graduado en la Universidad de Ciencias Informáticas (UCI) como Ingeniero en Ciencias Informáticas, 2012

Trabajador de la Dirección para la Comercialización y los Negocios
Correo electrónico: larza@uci.cu
Teléfono: 837-3865

AGRADECIMIENTOS

A mi papá y mi mamá, por haber colocado en mí el ansia de aprender y crecer.

A mi abuela Mimi, por el empeño en mi educación.

A mis hermanos Yosvey, Yaimara y Rafael.

A mi abuelo Mario, siempre ahí, preocupado, y atento.

A mis tías Rosy y Neyda por apoyarme en mis sueños y a mis primos y primas por compartirlos.

A Juanela, Papo, por ser un gran apoyo durante todos estos años.

A mis tíos y tías de Cienfuegos

A mis amigos de la infancia y adolescencia Alfredo, Gleibis, Marianela, Leydy, Nore, por continuar cerca de mí.

A mis amigos de la UCI, Leisy, Andy, Ledian, Rafa, Ada, Frank, Oslando, Ani, Cherry, Mursu, Sergio, Lázaro, Aliet, Mayra Pombo, a mis amigos del 3106, a mi nuevo grupo del CPE.

A mi tribunal y a mi oponente por las enseñanzas.

A mi profesora Elsidania, por ser mi mejor recuerdo pedagógico.

A Sandra, por el impulso definitivo.

A Juana por el cariño.

A las chicas Lotti, Saily y Yani fuente inmensa de alegría y amistad.

A mis tutoras Dianela y Leanet, más que tutoras, hermanas.

A mi Madrina Nancy, por todo lo aprendido y por ser una fuente de sabiduría.

A madrina Amaya y a Rachel, por la hermandad.

A Eliza y Belkis.

A mi amor Lester.

A Cuba y Antony, por la compañía.

A Yemayá, Obbatalá, a los orishas.

RESUMEN

La Muestra Joven ICAIC, es un evento dedicado a difundir la obra audiovisual de los jóvenes y a estimular el conocimiento y la reflexión, así como potenciar el diálogo entre las diversas generaciones que se dedican a esta manifestación artística en nuestro país, atesora la información que se genera de cada una de las ediciones de dicho evento. En este proceso se manipula una considerable cantidad de información, donde la gestión de la misma resulta tediosa y engorrosa a la vez, debido a las diferentes fuentes y soportes en las que se encuentra, lo que equivale a pérdidas considerables de tiempo por el difícil acceso, organización y actualización de la misma.

El presente trabajo recoge los resultados del desarrollo de un sistema automatizado mediante el cual se logra informatizar el proceso de gestión de la información asociado a las ediciones de la Muestra Joven ICAIC, contribuyendo a la organización de la información, al ahorro de tiempo y esfuerzo. Para lograr los objetivos se realizó un estudio de los referentes teóricos y metodológicos, así como de sistemas similares que valieron de ayuda para el desarrollo de la solución. Se siguieron los pasos que propone la metodología ágil de desarrollo de *software* Scrum. Se desarrolló sobre el IDE PhpStorm, usando los lenguajes PHP del lado del servidor y JavaScript, HTML, CSS del lado del cliente, como *frameworks* Symfony y Bootstrap. Como servidor web Apache y como gestor de base de datos PostgreSQL.

PALABRAS CLAVE

Información, gestión de información, sistemas de información, Sistemas de Gestión de Información.

ABSTRACT

The Muestra Joven ICAIC is an event dedicated to spreading the audiovisual work of young people and encourage knowledge and reflection, as well as maximize the dialogue between the different generations dedicated to this art in our country. The Muestra Joven ICAIC hoards the information produced in each one of the editions of the event. In this process, a considerable amount of information is handled and its management turns tedious, due to the different sources and media in which is stored. The difficult access to the information, its organization and actualization leads to considerable losses of time.

This study presents the results of the development of an automated system by which it's computerized the process of information management associated with this event, contributing to the organization of information, saving time and effort. A study of the theoretical and methodological references, and similar systems was made to achieve the objectives. The steps proposed by Scrum agile development methodology were followed. It was developed on PhpStorm IDE, using PHP as server side language, and JavaScript, HTML and CSS as client side language. The frameworks used were Symfony and Bootstrap. As web server was used Apache, and as database manager, PostgreSQL.

KEYWORDS

Information Management, Information Systems, Management Systems

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En los nuevos modelos de negocio la gestión de la información, la documentación y el conocimiento se perfilan como componentes estratégicos de primera magnitud. La orientación tradicional del carácter táctico de los proyectos de gestión de la información o de la documentación cambia cuando se considera un componente de la estrategia empresarial.

La información desde esta perspectiva se ha convertido en el activo principal de cualquier institución, representando en la mayoría de los casos su principal ventaja estratégica en su salvaguarda, para su uso y posterior recuperación.

Los Sistemas de Gestión de Información (SGI) permiten la gestión de la información de los recursos tanto internos como externos. Su finalidad es generar servicios y productos que respondan a las necesidades y sobrepasen las expectativas de los usuarios, posibilitando que el sistema trabaje eficiente y económicamente a la vez (1).

Es por ello que el desarrollo de SGI se hace cada vez más necesario en este sentido, ya que tienen la capacidad para almacenar datos, producir información y facilitarla para ser recuperada por los usuarios del sistema, para el proceso de toma de decisiones, lo cual es un atributo clave para este tipo de sistemas.

El ICAIC (Instituto Cubano de Arte e Industria Cinematográficos) entre los programas que desarrolla tiene la Muestra Joven ICAIC, el cual es un evento organizado y dirigido por el propio ICAIC y el MINCULT (Ministerio de Cultura de la República de Cuba) desde hace ya 15 años. En la actualidad cuenta con 14 ediciones y se realizó por primera vez del 30 de octubre al 3 de noviembre de 2000. Desde la 11na edición se realiza la primera semana de abril, de martes a domingo (2).

Es un evento dedicado a difundir la obra audiovisual de los jóvenes y a estimular el conocimiento y la reflexión alrededor de la misma, así como potenciar el diálogo entre las diversas generaciones que se dedican a esta manifestación artística en nuestro país. Reúne a jóvenes realizadores, guionistas, fotógrafos, productores, editores, sonidistas, diseñadores, entre otros especialistas (2).

Su diseño consta de una muestra principal, donde son exhibidas obras de ficción, documental y animación que han sido seleccionadas, previamente, para concurso y sus realizadores son jóvenes de hasta 35 años. La muestra auspicia un concurso de carteles, con el objetivo de estimular nuevas miradas hacia el quehacer

de la cartelística serigráfica nacional, donde participan estudiantes del ISDi (Instituto Superior de Diseño), así como otros diseñadores, quienes presentaran carteles a partir de las obras en concurso (2).

Haciendo cine, es otra de las secciones fundamentales dentro del evento que tiene 7 años de creada y constituye una alternativa para el fomento y apoyo a las producciones de ficción y documental de jóvenes realizadores. Este espacio además de favorecer la materialización de proyectos de ficción y documental, capacita y entrena a los más jóvenes en la modalidad de presentación de proyectos (*pitching*) (2).

También cuenta con espacios colaterales que se diseñan a partir de la concepción de cada edición: muestras homenajes, muestras internacionales, clases magistrales, conferencias, talleres, exposiciones, presentaciones de publicaciones y espacios de debates con público y críticos de las obras en concurso (2).

En la entrevista realizada, al organizador de este evento, se puede constatar que actualmente la información perteneciente a las pasadas ediciones de la Muestra Joven ICAIC se encuentra dispersa y en varios formatos: bases de datos realizadas en Access, en fichas técnicas, catálogos, discos duros, lo que hace el proceso de consulta por parte de los usuarios sumamente engorroso, y en ocasiones hay que apelar a la memoria, sin contar que la videoteca, con que cuenta la oficina de la muestra, no tiene diseñado un índice correctamente estructurado, que permita buscar una película por su título, género, director, guionista, entre otros datos.

Como se puede evidenciar la acumulación de información y fichas técnicas de las ediciones de la Muestra Joven ICAIC impiden el análisis de sus datos, la evolución de sus participantes, la sistematización de sus memorias e impide el estudio y análisis en perspectiva del evento y sus logros, además, genera pérdidas considerables de tiempo empleado en la obtención de la información; poca dinámica, debido a la ausencia de un proceso automatizado, a partir del cual se gestione la información de las ediciones, contribuya al mejoramiento del registro, organización, acceso, actualización y almacenamiento de la información generada para su uso y posterior recuperación.

Por lo anteriormente expuesto se describe el siguiente **problema a resolver**: Las insuficiencias en los procedimientos y carencia de herramientas informáticas, afectan la gestión de información de las ediciones de la Muestra Joven ICAIC.

Como **objeto de estudio:** Sistemas de Gestión de Información y como **campo de acción:** El proceso de recopilación de los recursos de información de las ediciones de la Muestra Joven ICAIC.

Para dar solución al problema planteado se propone como **objetivo general***:* Desarrollar un Sistema de Gestión de Información, que estructure la información dispersa y contribuya a la automatización del proceso de gestión de las ediciones de la Muestra Joven ICAIC.

Para dar cumplimiento al objetivo general se definieron las siguientes **tareas investigativas**:

- Estudio del estado del arte relativo a los SGI, metodologías de desarrollo, técnicas y herramientas de *software*.
- Identificación de los requisitos funcionales y no funcionales.
- Diseño de la solución propuesta.
- Implementación de la solución propuesta.
- Validación del sistema mediantes métricas Tamaño Operacional de Clases y Relaciones entre Clases.
- Análisis de los resultados.

Con el desarrollo de la solución se arribará a los siguientes resultados:

- Informe detallado con la base teórica-práctica sobre la cual se sustenta dicha solución.
- Sistema de Gestión de Información para la Muestra Joven ICAIC.

Para el desarrollo de las **tareas de investigación** se combinaron diferentes métodos y técnicas de la investigación científica para la búsqueda y procesamiento de la información, entre los que se encuentran:

Métodos teóricos:

Permiten estudiar las características del objeto de investigación que no son observables directamente, facilitan la construcción de modelos e hipótesis y posibilitan el conocimiento del estado del arte del fenómeno y su relación con otros fenómenos (3). Los utilizados para la realización de esta investigación son los siguientes:

- **Analítico-Sintético:** Mediante este método se descompone un objeto, fenómeno o proceso en los principales elementos que lo integran para analizar, valorar y conocer sus particularidades (4). En

este caso se utilizó para el análisis de las teorías, documentos y la extracción de los elementos más importantes que se relacionan con la Muestra Joven ICAIC; dando paso al desarrollo del sistema según las necesidades generadas y la recopilación de la bibliografía a utilizar.

- **Histórico-Lógico:** A través de este método se establece la necesaria correspondencia entre los elementos de los métodos lógico e histórico, proyectando el análisis de la evolución histórica de los fenómenos, con la proyección lógica de su comportamiento futuro (4). Este método es utilizado para saber, con mayor profundidad, los antecedentes y las tendencias actuales referidas a los SGI, conociendo así su trayectoria histórica a través de su origen y las herramientas para su desarrollo.

Métodos empíricos:

Estos métodos posibilitan revelar las relaciones esenciales y las características fundamentales del objeto de estudio, accesibles a la detección de la percepción, a través de procedimientos prácticos con el objeto y diversos medios de estudio (5).

- **Observación:** Es un método para reunir información visual sobre lo que ocurre, lo que el objeto de estudio hace o cómo se comporta (6). En este caso se utilizó para entender cómo se realiza hoy en la oficina de la Muestra Joven ICAIC el proceso de gestión de información.
- **Análisis de documentos:** Se estudiaron documentos relacionados con los SGI.

Técnicas de recopilación de información:

La recopilación de datos se utiliza para verificar los métodos empleados en lo investigado, para llegar a la conclusión del suceso, teniendo las pruebas y una serie de pasos que se llevan a cabo para comprobar la hipótesis planteada (7).

- **Entrevista:** Es una técnica para obtener datos, que consiste en un diálogo entre dos personas (7). Con su utilización se establece una comunicación con el personal encargado de dirigir y gestionar la Muestra Joven ICAIC. Para fundar la problemática planteada se recopila toda la información necesaria sobre la gestión de datos en la muestra de jóvenes realizadores del ICAIC.

La presente investigación queda estructurada en tres capítulos:

Capítulo 1. Sistemas de Gestión de Información.

Se realiza el análisis del estado del arte de la investigación. Se estudian las principales tendencias y tecnologías actuales, metodología de desarrollo del *software* y herramientas sobre las cuales se apoya la propuesta del sistema a desarrollar, dejando sentadas las bases teóricas de la investigación.

Capítulo 2. Características del sistema.

Se abarca el desarrollo de la fase de análisis y diseño del sistema, englobando todo lo referente al modelado del diseño. En el mismo se describen los requisitos y figuran los diagramas de clases del diseño de la solución propuesta, así como el modelo de dominio correspondiente.

Capítulo 3. Implementación y validación de la solución propuesta.

Se ofrecen los detalles relacionados con la implementación, mediante diagramas, de las funcionalidades definidas en el capítulo anterior. Se abordan los tipos de métricas realizadas para la validación del sistema y los resultados obtenidos al aplicar las mismas.

CAPÍTULO 1: SISTEMAS DE GESTIÓN DE INFORMACIÓN. 1

En el presente capítulo se analizan los referentes teóricos y metodológicos relacionados con la investigación. Se hace un estudio de sistemas similares que puedan servir de ayuda en la construcción de la solución. Además de la selección, mediante un estudio detallado, de las tecnologías y herramientas adecuadas para el desarrollo de la solución.

1.1 Tendencias y enfoques de la gestión de la información.

Las tendencias que marcan la gestión de empresas u organizaciones llevan implícito un aspecto fundamental, que en muchas ocasiones no se considera, como lo es la información, sus sistemas y procesos de gestión.

En la actualidad, la información empieza a considerarse como un recurso económico debido a que una vez procesada y utilizada racionalmente, adquiere valor, sobre todo si su utilidad se ubica en función de su aporte al proceso de toma de decisiones, creación de productos y solución de problemas, entre otros aspectos.

Aún con la habilidad de manipular el vasto volumen de datos y la velocidad con que crece, la información tiende a transformarse en una distracción con una enorme responsabilidad. Esta paradoja maneja la necesidad de incrementar el poder y la flexibilidad de los SGI. Para obtener los grandes y complejos datos, los usuarios deben tener herramientas que les simplifiquen las tareas de manipulación de datos y extracción de información en tiempo, en forma segura y confiable (8).

La información desempeña un papel estratégico para determinar la capacidad de una organización para crecer y adaptarse. En primer lugar, la organización utiliza la información para recibir cambios y desarrollos en su medio externo. El segundo campo de uso estratégico de la información es cuando las organizaciones crean, organizan y procesan información a fin de generar nuevos conocimientos a través del aprendizaje organizacional. El tercer campo de uso estratégico de la información es cuando las organizaciones buscan y evalúan información a fin de tomar decisiones importantes (9).

A modo de generalización, la siguiente definición puede sintetizar lo abordado anteriormente y a la cual se adscribe el autor… "La información puede entenderse como la significación que adquieren los datos como resultado de un proceso consciente e intencional de adecuación de tres elementos: los datos del entorno, los propósitos y el contexto de aplicación, así como la estructura de conocimiento del sujeto.” (10).

Pero gestionarla adecuadamente requiere identificar la información relevante, conocer dónde y cómo obtenerla, y disponer de herramientas y recursos necesarios para su tratamiento, difusión y puesta en valor. Todo esto no se puede hacer de manera improvisada, para ello se necesita planificar el SGI.

La gestión de información surge como un nuevo concepto dentro del campo de la ciencia de la información, orientado al manejo de la inteligencia corporativa de una organización, que permite la estructuración interna a las organizaciones y les permite reaccionar ante los cambios de su entorno, apoyándose en el uso de la información y de los recursos de información disponibles.

El autor (11), denomina la gestión de la información como una triple hélice. En primer lugar lo importante de la información es su contenido y no tanto su soporte, en segundo lugar considera que los gastos para sistemas y tecnologías de la información son un gasto para recursos y no deben ser considerados como gastos generales de funcionamiento, y la tercera parte, de la filosofía de la gestión de la información, es la exacta coordinación del recurso dentro de la propia organización, debido a que en la actualidad este recurso está disperso en muchas empresas.

Según (12), el concepto de gestión de recursos de información, (viendo la información como un recurso), se define como el manejo de la inteligencia corporativa de una organización a objeto de incrementar y mejorar el cumplimiento de sus metas.

Por tanto, la gestión de recursos de información, se lleva a cabo mediante la planificación, organización, ejecución y control de los recursos, responsables de crear, explotar y desarrollar el sistema de información integral, que permita el manejo de la información organizacional en función de la consecución de los objetivos de la organización.

De lo anterior se puede inferir que la gestión de información es más general, y está vinculada con la generación y aplicación de estrategias, políticas y teniendo en cuenta, la cultura organizacional, para lograr un mejor uso de la información.

Según (13) “La gestión moderna está orientada a producir o brindar servicios que respondan a las necesidades de las personas y particularmente a sus clientes/usuarios”, lo que constituye el enfoque fundamental que conduce al cambio.

La profunda revolución tecnológica de la que todavía no se tiene perspectiva suficiente para saber a dónde llegará, ha sido el motor de este cambio. Por esta razón muchas veces los empresarios y directivos simplifican su actuación frente a la nueva realidad, centrándola en la compra e instalación de herramientas informáticas de última generación que deberían dar resultados a corto plazo (14).

Obtener un resultado de las tecnologías de la información dependerá de cuán inteligentemente se gestionen. Y parte de esa inteligencia consiste en pasar a entender que la función de las tecnologías, dígase también sistemas, de información es gestionar mejor la información, para convertirla en conocimiento, personal u organizacional.

Los sistemas de información presentan un amplio abanico de definiciones, variando según los autores analizados en la investigación realizada en (15), definiéndolos en cuatro grupos como Conjunto, Sistema, Colección y Combinación.

En (16) se especifica que los sistemas de información son un conjunto organizado de personas, procesos y recursos, incluyendo la información y sus tecnologías asociadas, que interactúan de forma dinámica, para satisfacer las necesidades informativas que posibilitan alcanzar los objetivos de una o varias organizaciones.

Otro punto de vista, al definir sistemas de información, es el abordado por (17) cuando plantea que es una combinación organizada de personas, *hardware*, *software*, redes de comunicaciones y recursos de datos que reúne, transforma y disemina información en una organización.

Los sistemas de información canalizan la información desde las fuentes a los receptores. Sus funciones principales son conservar, custodiar, almacenar, identificar, representar, y difundir los recursos. Tipos de sistemas de información son los archivos, bases de datos, bibliotecas, documentación administrativa, mediatecas, museos y Sistemas de Gestión de Información.

Por tanto un sistema de información se refiere a los métodos, medios, materiales, generadores, contenedores y universo de involucrados en una forma organizada para efectuar la transferencia de información dentro de una actividad, campo u organización.

Autores como (18), (21) definen sistemas de información como el conjunto de procesos que, operando sobre una colección de datos estructurada de acuerdo a una empresa, recopila, elabora y distribuye la información necesaria para la información de dicha empresa y para las actividades de dirección y control correspondientes, apoyando al menos en parte, la toma de decisiones necesarias para desempeñar las funciones y procesos de negocios de la empresa de acuerdo a su estrategia, una vez recuperada esta información. En la figura a continuación se observa una muestra de ello.

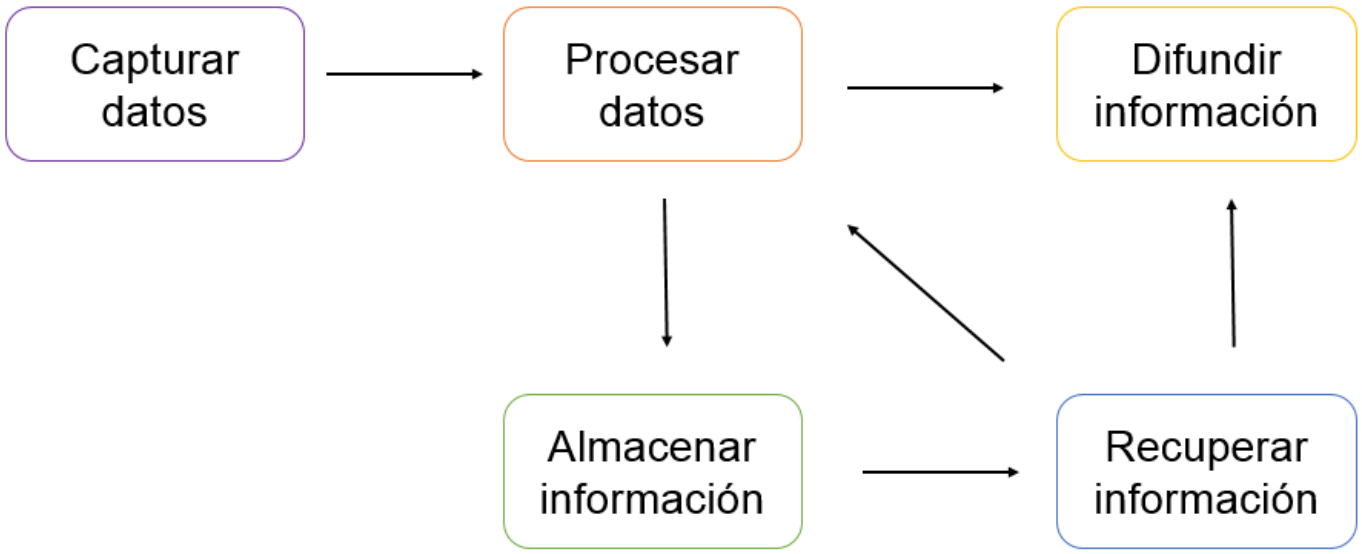

Fig. 1 Sistema de información. Fuente (Elaboración propia).

El contar con modernas estructuras de tecnología informática (*hardware*, comunicaciones y *software*) no garantiza a una empresa estar gestionando la información que ella misma genera eficazmente, promoviendo su mejora y crecimiento como una organización que aprende.

Por lo antes expuesto el autor considera que a pesar de los avances de la tecnología informática en materia de gestión de información se puede observar que se presenta la paradoja de empresas muy ricas en datos y muy pobres en información. La “riqueza” radica en los millones de registros acumulados a lo largo de los

años. La “pobreza”, en que no cuentan con las herramientas para explotar estos datos, convertirlos en conocimiento y mucho menos que este conocimiento esté disponible para toda la organización.

1.2 Sistemas de Gestión de Información y procesos asociados.

Los Sistemas de Gestión de la Información contienen un conjunto integrado de procesos, principalmente formales, que la organización conoce y sabe cómo utilizar (los informales no están excluidos), y son registrados en datos, a través de una base de datos. Se desarrollan en un entorno usuario-ordenador, sobre un conjunto de datos estructurados (base de datos), utilizando *hardware* y *software* computacional, redes de telecomunicaciones, técnicas de administración u otras formas de tecnología de información (19).

Se caracterizan por: tener la información disponible cuando es necesaria y con los medios adecuados, el suministro de la información de manera selectiva (cantidad contra calidad), variedad en las formas de presentación de la información (gráfica, numérica, entre otras), el grado de inteligencia del sistema, tiempo de respuesta del sistema (desde una petición hasta su realización), exactitud (conformidad entre los datos suministrados y los reales), generalidad (disponibilidad para atender diferentes necesidades), flexibilidad (capacidad de adaptación a nuevas necesidades), fiabilidad (probabilidad de operatividad correcta durante un período determinado de uso), seguridad (protección contra pérdida y/o uso no autorizado de recursos), reserva (nivel de repetición de la información para proteger las pérdidas), y amigabilidad (necesidad de aprendizaje para su manejo) (19).

Un sistema moderno de gestión de información exige la aplicación de nuevas tecnologías de información y comunicación; sin embargo, la tecnología por sí sola no es suficiente para lograr una buena gestión de información. Son diversos los procesos que conforman los Sistemas de Gestión de Información; ellos generan las entradas y salidas del sistema o de otros procesos relacionados; también pueden identificarse, controlarse, corregirse o actualizarse en la medida en que se producen las transformaciones del entorno y evoluciona la organización, como vía incuestionable para garantizar su calidad, eficiencia y mejora continua (19).

Un sistema de gestión es un conjunto de etapas unidas en un proceso continuo, que permite trabajar ordenadamente una idea hasta lograr mejoras y su continuidad.

Autores como (20) establecen que las etapas por las que transita el proceso de recopilación de información para la construcción de Sistemas de Gestión de Información son:

1. Etapa de Ideación: Identificar las necesidades de información crítica para la organización.
2. Etapa de Planeación: Reconocer y sistematizar los procesos de gestión de la información, así como la planeación de los mismos para su ejecución, con el apoyo de metodologías que así lo permitan.
3. Etapa de Implementación: Conectar la gestión de la información, entendida como un proceso integral y no aislado, con el resto de procesos clave de la organización y establecer los flujos de información para detectar problemas (“nichos informativos”, “*gaps*”, entre otros) y desarrollar las acciones necesarias para su solución.
4. Etapa de Control: Llevar a cabo el control de cada una de las etapas en la concepción del sistema por hitos de ejecución para evaluar los procesos de gestión.

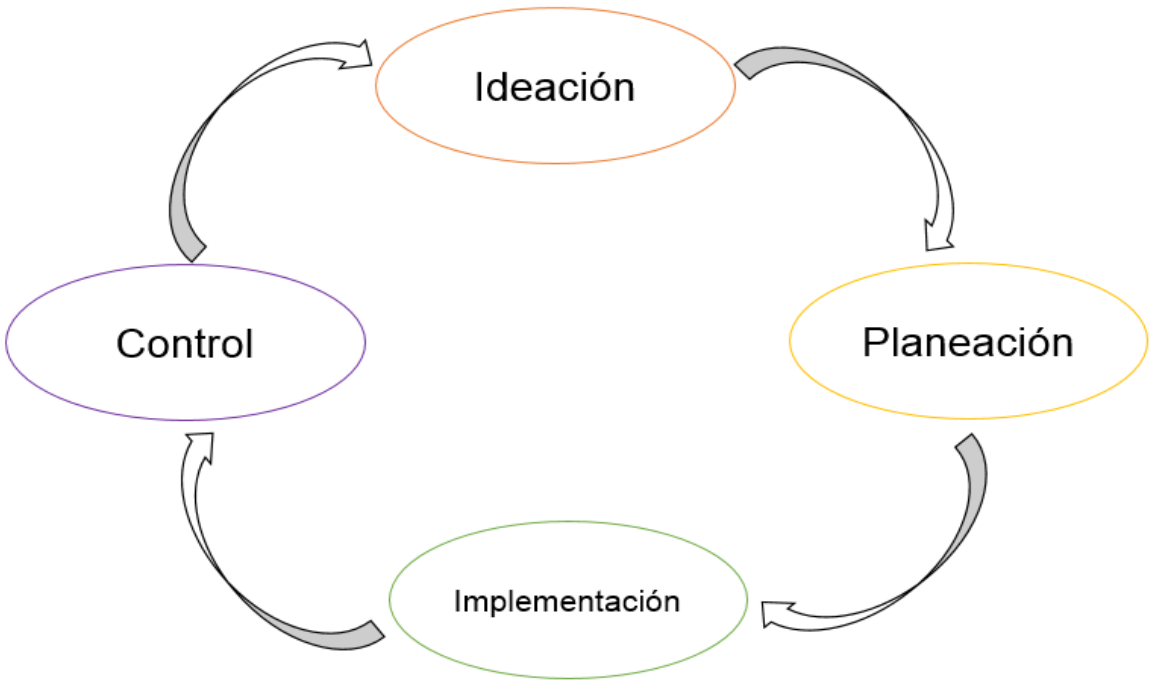

Fig. 2 Etapas del Sistema de Gestión. Fuente: (Elaboración propia).

Como se puede observar es un proceso circular, pues en la medida que el ciclo se repita recurrente y recursivamente, se logrará en cada ciclo, obtener mejoras.

Este estudio permitió al autor profundizar en la organización de los procesos de información con respecto a las áreas críticas de la muestra y en consecuencia aplicar estas etapas de gestión a lo largo del desarrollo de la investigación; teniendo un primer ciclo donde: se realizan encuentros con los organizadores de la Muestra Joven ICAIC para lograr identificar sus necesidades, se decide utilizar una metodología (la elección de la misma se detalla en el epígrafe 1.3.1) para guiar el resto del desarrollo, se especifican los requerimientos, se elabora un Modelo de Dominio del negocio en cuestión y se concluye, conjuntamente al cliente, en una discusión sobre dicho dominio. Los siguientes ciclos se evidencian mediantes las iteraciones propuestas por la metodología utilizada.

Un sistema de información automatizado o basado en computadoras, es la integración de *hardware*, *software*, personas, procedimientos y datos, todos estos componentes en función de garantizar que las empresas lleven a cabo sus tareas con mayor calidad y facilidad (21).

El acelerado crecimiento de las redes de computadoras y el desarrollo de aplicaciones web se ha convertido en una vía de vital importancia para la gestión de la información.

Una aplicación web es simplemente un programa informático que se ha de acceder mediante una conexión a internet o una intranet, mediante un navegador web (22).

En la actualidad las aplicaciones web han tenido mucho éxito gracias a lo fácil que resulta usar un navegador web como cliente ligero, esto además supone otras ventajas como la independencia del sistema operativo que utilice el usuario. Otra razón de su popularidad es la facilidad para actualizarlas y mantenerlas, ya que no requiere la distribución, instalación y actualización de la aplicación en miles de usuarios; basta actualizar el servidor para contar con nuevas versiones de la aplicación (22).

Las aplicaciones pueden ser desde pequeñísimos programas de apenas unas líneas de código, hasta grandes obras de ingeniería informática, con miles de horas de trabajo detrás. El tamaño, sin embargo, no define el éxito de una aplicación, sino justamente que cubra las necesidades del usuario (23).

En consideración a lo antes mencionado, se decidió realizar un SGI mediante una aplicación web.

A modo de conclusión el autor considera que un SGI automatizado es la colaboración entre el hombre y una serie de artefactos que posibilitan de una manera más fácil y organizada manejar, recoger, procesar, almacenar y distribuir la información.

1.2.1 Análisis de Sistemas de Gestión de Información existentes.

La bibliografía consultada sobre SGI, no arrojó resultados que resuelvan el problema en su totalidad, lo que acorta la posibilidad de encontrar una aplicación que sea reutilizable para poder dar solución al problema que se presenta en esta investigación.

Es válido analizar algunos sistemas de gestión, con el objetivo de obtener una visión de cómo se manejan algunos de los procesos asociados a los Sistemas de Gestión de Información en general y tomar ideas para la construcción de la solución.

Actualmente en la Universidad de las Ciencias Informáticas existe una aplicación (GATSERVER) (24), en la infraestructura de Soporte Técnico, que facilita la gestión de los reportes de solicitud de mantenimiento en la universidad. La aplicación está enfocada sólo a los servicios técnicos que se prestan a las distintas estructuras y básicamente a la gestión de los reportes que se derivan de esa actividad, o sea que no se tiene una base de datos o inventario de la tecnología existente en las áreas, solo la reparación y mantenimiento de la misma.

Como se puede apreciar esta aplicación no es factible, pues no cumple con los requisitos necesarios que exige el proceso a informatizar.

Otro *software* que permite gran número de funcionalidades enfocadas a la gestión de la información es la Plataforma de Acceso a Información (*Information Access Platform*, IAP por sus siglas en inglés) de ZyIMAGE, que ofrece a las organizaciones la mejor gama de herramientas para archivar, buscar, gestionar y compartir información electrónica y en papel, así como mensajes de correo electrónico y multimedia (25).

La construcción abierta y modular de ZyIMAGE permite que las soluciones se puedan configurar para adaptarse a los presupuestos y necesidades reales de los clientes, desde las instalaciones más básicas, a las soluciones empresariales exhaustivas (25).

OrangeHRM está dirigido hacia empresas de pequeño y mediano tamaño para gestión y administración del personal de las mismas. Es un *software* de código libre. Se basa en una solución de Apache, PHP y MySql. Está organizado de forma modular, estos módulos son: Administración, Administración de Información Personal (*Personal Information Manager*, PIM por sus siglas en inglés) y Reportes (26).

Desde el módulo de Administración se podrá visualizar información de los otros módulos, generar reportes y visualizar a cada uno de los trabajadores de la organización. En el módulo PIM se podrá administrar el registro del personal de la empresa. Permite la introducción de las características: horas de trabajo, fecha de ingreso y datos personales. En el módulo de Reportes es donde se concentran todos los reportes requeridos y necesarios de los datos que se tienen registrados (26).

Este sistema tiene como aspecto negativo la falta de documentación referente a su utilización, cuenta con una ayuda muy pequeña y confusa.

ASSETS NS es un sistema de gestión integral, una aplicación cliente-servidor programada en Visual Basic 6.0 y Microsoft SQL Server 2000. Es un sistema estándar y parametrizado que permite el control de los procesos de compras, ventas, producción, taller, inventario, finanzas, contabilidad, presupuesto, activos fijos, útiles y herramientas (27).

Como aspecto negativo de este *software* se puede mencionar que no es multiplataforma y de código abierto, así como que no garantiza la independencia tecnológica debido a que está desarrollado con tecnologías privativas.

A pesar de que estos sistemas pueden resultar altamente eficientes en empresas que lo utilicen, no son factibles como parte de la solución que se desea obtener, ya que son sistemas hechos a la medida, y su reusabilidad, en el negocio en cuestión, está limitada por las especificidades propias del entorno para el cual fueron construidas.

Por tanto, el autor considera que es poco factible adquirir un sistema ya desarrollado para un esquema que habría que personalizar totalmente de acuerdo a las características de la institución, y del tipo de información a procesar, en este caso de la Muestra Joven ICAIC. Es por ello que a pesar de la existencia de disímiles sistemas con funcionalidades altamente eficientes dentro del marco que fueron desarrollados, su uso no es extensible al campo de acción de esta investigación.

Sin embargo, su estudio fue provechoso, ya que las características y funcionalidades de estos sistemas sirven de base para el desarrollo del sistema de gestión de información propuesto.

1.3 Metodología, herramientas y tecnologías a utilizar.

Para el desarrollo de un *software* se deben tener en cuenta ciertos elementos, como son la metodología, tecnologías y las herramientas básicas para su construcción. En Cuba existe una política referente al uso de herramientas libres debido a la imposibilidad de adquirir *software* con licencias privadas, por lo que se han analizado una serie de herramientas que respondan a esta condición, además de lenguajes de programación y metodologías de desarrollo de *software*. A continuación se presenta el entorno tecnológico definido por el autor para darle solución a la problemática planteada a lo largo de esta investigación.

1.3.1 Metodología de desarrollo de software.

"Desarrollar un buen *software* depende de un gran número de actividades y etapas", así opina (28) y continua su reflexión, "donde el impacto de elegir la mejor metodología para un equipo, en un determinado proyecto, es trascendental para el éxito del producto". Este autor aclara en su investigación que el uso de las metodologías para el desarrollo de un *software* es esencial en un proyecto, la cual debe encajar en el equipo, guiar y organizar actividades que conlleven a las metas trazadas en el grupo.

La investigación realizada por (29) se encuentra centrada en aquellas metodologías más tradicionales centralizadas especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán.

Este tipo de metodologías tienen a su favor que proveen al equipo de un alto grado de ordenamiento, de disciplina, pero al no adaptarse apropiadamente a los cambios no son métodos adecuados cuando se trabaja en un entorno donde los requisitos no pueden predecirse o pueden variar, estas metodologías se centran más en los procesos que en las personas, el cliente puede llegar a ser relegado. Estas propuestas han demostrado ser efectivas y necesarias en un gran número de proyectos, pero también han presentado problemas en otros muchos.

Por lo que se presentan las metodologías ágiles centradas, según (30), en el factor humano o en el producto del *software*. Las mismas están mostrando su efectividad fundamentalmente en proyectos que presentan requisitos cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero sin alterar la calidad.

Las necesidades del cliente pueden variar desde el momento de contratación de un *software* al momento de su entrega; y es de mayor importancia satisfacer estas últimas que las primeras. Esto requiere procesos de *software* que en lugar de rechazar los cambios los incorpore. Las metodologías ágiles suministran una serie de normas y principios junto a técnicas que intentan hacer la entrega del proyecto menos complicada y más satisfactoria tanto para los clientes como para los equipos de entrega.

Se ha definido para el desarrollo del sistema en cuestión utilizar como guía la propuesta de una metodología ágil, debido a que permite principalmente disminuir costos, satisfacer las necesidades del cliente y la variación de los requisitos en el tiempo. Además de presentar un equipo de desarrollo pequeño y poder hacer al cliente parte del mismo.

A continuación se detallan las principales características de dos de las metodologías ágiles más frecuentemente usadas en el mundo del desarrollo de *software*, de las cuales se elegirá una de ellas para guiar el proceso de desarrollo del sistema en cuestión.

XP (*eXtreme Programming*, por sus siglas en inglés), típicamente esta metodología requiere de 10 a 15 ciclos o iteraciones, así plantea (31), los cambios que se realizan en los requisitos son un aspecto natural del desarrollo de los proyectos, por lo que hace que la adaptabilidad tenga un mayor impacto frente a la previsión en el diseño del *software*.

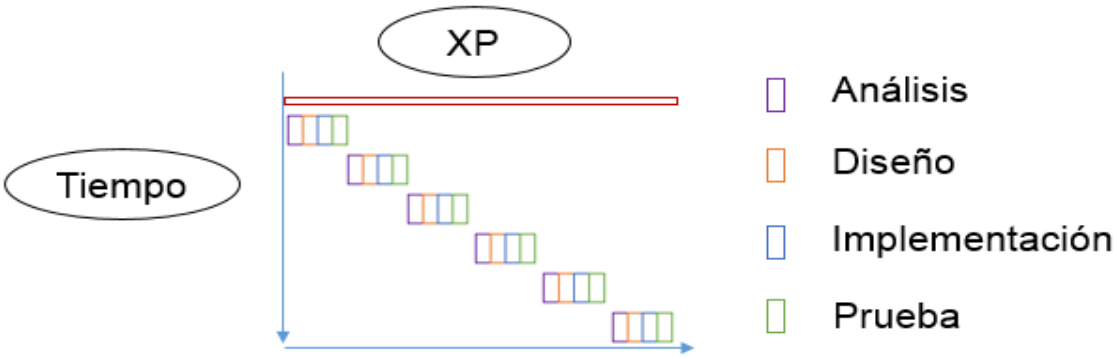

Fig. 3 Ciclos XP. Fuente (Elaboración propia).

La metodología XP plantea la planificación como un diálogo continuo entre las partes involucradas en el proyecto, así se refleja en la investigación realizada por (31), y continua, incluyendo al cliente, a los

programadores y a los coordinadores o gerentes. El proyecto comienza recopilando “historias de usuarios”, las que sustituyen a los tradicionales “casos de uso”. Una vez obtenidas las “historias de usuarios”, los programadores evalúan rápidamente el tiempo de desarrollo de cada una; si alguna de ellas tiene “riesgos” que no establecen con certeza la complejidad del desarrollo, se realizan pequeños programas de prueba (*spikes*), para reducir estos riesgos. Una vez realizadas estas estimaciones, se organiza una reunión de planificación, con los diversos actores del proyecto (cliente, desarrolladores, gerentes), a los efectos de establecer un plan o cronograma de entregas (*Release Plan*) en los que todos estén de acuerdo. Una vez acordado este cronograma, comienza una fase de iteraciones, en donde en cada una de ellas se desarrolla, prueba e instala unas pocas “historias de usuarios”.

Las características fundamentales de esta metodología son (28):

- **Desarrollo iterativo e incremental:** pequeñas mejoras, unas tras otras.
- **Pruebas unitarias continuas**: frecuentemente repetidas y automatizadas, incluyendo regresión de pruebas. Se aconseja escribir el código de la prueba antes de la codificación.
- **Programación por parejas**: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que se obtiene mayor calidad del código escrito pues el código es revisado y discutido mientras se escribe. Es más importante que la posible pérdida de productividad inmediata.
- **Frecuente interacción del equipo de programación con el cliente o usuario:** se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- **Corrección:** de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- **Refactorización del código:** reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- **Propiedad del código compartida**: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

- **Simplicidad en el código**: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

Scrum es una metodología ágil de desarrollo de *software,* que permite tener un control continuo sobre el estado actual del *software*. El cliente establece las prioridades y el equipo se auto-organiza para determinar la mejor forma de entregar los requerimientos con más prioridades. Scrum implica una filosofía de trabajo que no sólo requiere al desarrollador sino también al cliente, dando prioridad a los individuos y las interacciones sobre los procesos y las tareas, prefiriendo el *software* funcional sobre la excesiva documentación (32).

En Scrum un proyecto se ejecuta en bloques temporales (iteraciones-*sprint*) de un mes natural (pueden ser de dos o tres semanas, si así se necesita). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto que sea susceptible de ser entregado con el mínimo esfuerzo cuando el cliente lo solicite (31).

Scrum se puede dividir de forma general en 3 fases, las cuales se entienden como reuniones: Planificación del *Backlog* donde se define el documento que refleja los requisitos del sistema organizado por prioridades, además de la planificación del *Sprint* 0, en la que se decide los objetivos y el trabajo para esta iteración, el objetivo más importante de esta fase es obtener el *Sprint Backlog*, siendo esta la lista de tareas a realizar; Seguimiento del *Sprint*, en esta fase se realizan reuniones diarias para evaluar el avance de las tareas; y por último Revisión del *Sprint*, en esta fase se realiza una revisión del incremento generado, se presentan los resultados finales y una versión (33).

Fig. 4 Ciclo de Scrum. *Sprint*. Fuente (Elaboración propia).

Los elementos que forman Scrum son (33):

- ✓ *Product Backlog* (Pila de Productos): Lista de necesidades del cliente.
- ✓ *Sprint Backlog* (Pila de *Sprint*): Lista de tareas que se realizan en un *sprint*.
- ✓ Incremento: Parte añadida o desarrollada en un *sprint*, es una parte terminada y totalmente operativa.

Esta metodología por su proceso iterativo incremental produce un grupo de funcionalidades en cada fin de iteración (29):

- Proceso ágil para el manejo y control del trabajo de desarrollo.
- Contenedor de prácticas de ingeniería existentes.

- Enfoque basado en equipos, incrementa el desarrollo cuando los requerimientos cambian rápidamente.
- Proceso que controla el caos entre los conflictos de interés y las necesidades.
- Camino para mejorar las comunicaciones y maximizar la cooperación.
- Camino para detectar la causa y solucionar cualquier problema en el desarrollo.
- Escalable desde proyectos simples a proyectos completos organizacionales, Scrum ha controlado y organizado el desarrollo de productos y proyectos con miles de desarrolladores e implementadores.
- Ruta para sentirse bien en el trabajo.

Sus principales ventajas son (34):

- Se trabaja en iteraciones cortas, de alto enfoque y total transparencia.
- Se acepta que el cambio es una constante universal y se adapta el desarrollo para integrar los cambios que son importantes.
- Se incentiva la creatividad de los desarrolladores haciendo que el equipo sea auto administrado.
- Se mantiene la efectividad del equipo habilitando y protegiendo un entorno libre de interrupciones e interferencias.
- Permite producir *software* de una forma consistente, sostenida y competitiva.

Fundamentos de la selección:

Se decide utilizar en el desarrollo de la solución propuesta la metodología de desarrollo Scrum debido a que la misma es adecuada para entornos que se caracterizan por tener incertidumbre, es decir, el objetivo que se quiere alcanzar está planteado pero no se tiene un plan detallado del producto; auto-organización, este es el caso donde al ser el equipo de trabajo considerablemente pequeño es capaz de organizarse sin tener roles específicos, mostrando autonomía, auto-superación y auto-enriquecimiento; auto-control, el control debe ser moderado dejando espacio para la creatividad y la espontaneidad.

1.3.2 Lenguaje de programación.

Un lenguaje de programación es un lenguaje formal diseñado para expresar procesos que pueden ser llevados a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana (35).

Lenguaje de programación del lado del servidor.

Java es un lenguaje de programación frecuentemente usado por desarrolladores. Su sintaxis toma mucho de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria (36).

Este es un lenguaje multiplataforma y de distribución libre, lo que implica que no sea necesario pagar una licencia para comenzar a desarrollarlo, así mismo es un lenguaje muy completo y poderoso, pues posee librerías y utilidades muy completas que facilitan la programación, aunque tiene la desventaja de ser un lenguaje de ejecución lenta, debido al uso de la máquina virtual de Java (36).

PHP, acrónimo de "Procesador de Hipertexto *(Hypertext Preprocessor)*", es un lenguaje interpretado por el servidor generando un HTML1 (Lenguaje de Marcas de Hipertexto, por sus siglas en inglés *HyperText Markup Language*) con el resultado de sustituir las secuencias de instrucciones PHP por su salida. Ampliamente utilizado, de código abierto y de propósito general bajo licencia GPL (Licencia Pública General, por sus siglas en inglés *General Public License).* Es un lenguaje de *scripting* adecuado para el desarrollo web y embebido en páginas HTML. El principal objetivo del lenguaje es permitir que los desarrolladores web escriban páginas generadas dinámicamente de forma rápida, pero se puede hacer mucho más con PHP (37).

Entre sus principales características se destacan su potencia, su alto rendimiento, su facilidad de aprendizaje y su escasez de consumo de recursos, permitiendo al desarrollador realizar varias funciones, desde generar documentos en formato pdf hasta analizar código XML2 (Lenguaje de Marcas eXtensibles, *eXtensible Markup Language* por sus siglas en inglés). Es multiplataforma y con capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destacando su conectividad con MySQL, lo cual permite la creación de aplicaciones web robustas (37).

Fundamentos de la selección:

El autor considera que se seleccione PHP para el desarrollo de esta aplicación pues permite la creación de páginas dinámicas de forma más rápida y flexible y el envío de datos hacia los formularios del cliente, cuenta con mayor seguridad contra los distintos tipos de ataques. Permite darle respuesta a las peticiones de los clientes de manera rápida y segura. Se encuentra bajo licencia GPL lo que permite hacerle modificaciones a su código fuente.

Lenguaje de programación del lado del Cliente

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas, incorporando efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Los programas escritos con este lenguaje se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. Se trata de un lenguaje de programación del lado del cliente compatible con la mayoría de los navegadores modernos, por consiguiente permite la máxima interactividad entre el usuario y la página, además la verificación de los datos introducidos por el usuario antes de enviar el formulario al servidor, el manejo de *applets* y *plugins* dentro de múltiples marcos de HTML (38).

El navegador del cliente es el encargado de interpretar las instrucciones de JavaScript y ejecutarlas. Es un lenguaje con muchas posibilidades, permite la programación de pequeños *scripts*, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems (38).

HTML es un lenguaje de composición de documentos y especificación de ligas de hipertexto, que define la sintaxis y coloca instrucciones especiales que no muestra el navegador, aunque se le indica como desplegar el contenido del documento, incluyendo texto, imágenes y otros medios soportados. HTML también indica cómo hacer un documento interactivo a través de ligas especiales de hipertexto, las cuales conectan diferentes documentos ya sea en su computadora o en otras, así como otros recursos de Internet, como FTP (Protocolo de Transferencia de Archivos, *File Transfer Protocol* por sus siglas en inglés) y Gopher (39).

La descripción se basa en especificar la estructura lógica del contenido así como los diferentes efectos que se quieran dar. Se puede decir que de algún modo el HTML va en contra de los principios por el cual fue creado, ya que detalla mayormente la forma en la que se va a mostrar el contenido que la información o el mensaje incluido. Las páginas web luego pueden ser vistas por un tipo de aplicación llamada navegador o *browser* en inglés. El navegador muestra el texto y multimedia de acuerdo a la forma especificada o mejor interpretada, puesto a la diversidad de navegadores existentes, los cuales no son capaces de interpretar un mismo código de una manera unificada (39).

La evolución tan desordenada del HTML ha puesto toda una serie de inconvenientes y deficiencias, que para superarlos se han introducido otras tecnologías accesorias para organizar, optimizar, engalanar y automatizar el funcionamiento de la web como CSS y JavaScript por citar algunas (39).

También es importante conocer las limitaciones de este lenguaje. HTML no es una herramienta para procesar texto, una solución de edición electrónica ni tampoco un lenguaje de programación. Su propósito fundamental es definir la estructura y apariencia básica de documentos y conjunto de documentos de tal manera que puedan ser manejados de forma rápida y fácil por un usuario en la red para verlo en diferentes dispositivos (39).

Las hojas de estilo en cascada o CSS (*Cascading Style Sheets*) surgen por la limitación del lenguaje HTML a la hora de darle forma a un documento, permitiendo dar solución a estos problemas ya que son mecanismos simples que posibilitan aplicar formato a los documentos escritos en HTML y en lenguajes estructurados como XML, dando la posibilidad de separar el contenido de la presentación, es decir, la información estará incluida en la página HTML pero sin ser este el archivo que defina como será visualizada la información, las indicaciones referentes a cómo quedará comprendido visualmente el documento estarán especificadas en los archivos de la CSS (40).

Describe cómo se verá un documento en pantalla, siendo esta forma de descripción de estilos la vía que ofrece a desarrolladores el control total de estilos y formato de sus documentos así como de múltiples páginas web al mismo tiempo. Un cambio en el estilo para un elemento en la CSS implica que todas las páginas vinculadas a esa CSS en la que aparezca ese elemento serán afectadas (40).

Funciona a través de reglas, o sea declaraciones sobre el estilo de uno o más elementos. Las hojas de estilo están compuestas por varias de esas reglas aplicadas a un documento HTML o XML. Estas reglas

constan de dos partes: un selector y la declaración. La declaración está compuesta por una propiedad y el valor que se le asigne (40).

El selector es como el enlace entre el documento y el estilo, especificando los elementos que serán afectados por esa declaración. La declaración es la parte de la regla que establece cuál será el efecto. Una de las formas que se utilizan para dar estilo a un documento es la hoja de estilo externa, que se vincula al documento a través del elemento link, que debe estar situado en la sección <HEAD >, también utilizando el elemento <STYLE> dentro del documento que se le quiere dar estilo, y que generalmente se encuentra situado en la sección <HEAD>, siendo así los estilos reconocidos antes de que la página se cargue completamente (40).

Otra forma de dar estilo a un documento es utilizando estilos sobre los elementos que lo permiten a través del atributo <STYLE> dentro del <BODY>. Es importante resaltar el cuidado que se debe tener a la hora de usar esta tecnología ya que muchos usuarios puede que no vean los formatos que son aplicados a las páginas con CSS (40).

Fundamentos de la selección:

El autor considera que la combinación de HTML, JavaScript y CSS, como lenguajes del lado del cliente, beneficia el desarrollo de la aplicación ya que las limitaciones de uno son suplidas por el otro, permitiendo así obtener una solución dinámica y agradable a la vista del usuario.

1.3.3 Marco de trabajo.

En el desarrollo de *software*, un marco de trabajo o *framework, es* una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de *software* concretos, con base a la cual otro proyecto de *software* puede ser más fácilmente organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto (41).

Symfonny 2.3 es un *framework* diseñado para optimizar el desarrollo de aplicaciones web basándose en el patrón Modelo Vista Controlador. Este patrón de diseño permite separar la lógica del negocio, del servidor y la presentación de la aplicación. Cuenta con un número de clases y herramientas que facilitan la reducción del tiempo de desarrollo de una aplicación web de gran complejidad. Automatiza las tareas comunes, esto

proporciona al desarrollador la capacidad de dedicarse por completo a los aspectos específicos de cada aplicación. Este marco de trabajo es compatible con diversos gestores de base de datos, pudiendo mencionar entre ellos, MySQL, PostgreSQL, Oracle. Es multiplataforma, utiliza programación orientada a objetos, por lo que es indispensable la utilización de PHP como lenguaje de programación, es sencillo de usar, fácil de extender, presenta una potente línea de comandos ideal para la generación de código, lo que contribuye al ahorro de tiempo de trabajo (42).

Bootstrap 3.0 es un *framework* o conjunto de herramientas libres originalmente creado por Twitter, que permite simplificar la creación de diseños web utilizando hojas de estilo CSS y JavaScript. Tiene como característica la de poder adaptar la interfaz web a los distintos navegadores y al tamaño del dispositivo que se visualice. Esta técnica de diseño y desarrollo es conocida como diseño adaptativo (*responsive design*). Este *framework* ofrece numerosos componentes que ahorran tiempo y esfuerzo, siendo sus diseños simples, limpios e intuitivos. Se integra con las principales librerías de JavaScript, por ejemplo JQuery (43).

1.3.4 Entorno de Desarrollo Integrado

IDE (Entorno de Desarrollo Integrado, por sus siglas en inglés *Integrated Development Environment*), empaquetado como un programa de aplicación, o sea, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (44).

NetBeans 8.0 es un IDE que permite programar en distintos lenguajes, es ideal para trabajar con el lenguaje de desarrollo JAVA (y todos sus derivados), además ofrece un excelente entorno para programar en PHP. NetBeans es perfecto y muy cómodo para los programadores. Tiene un excelente balance entre una interfaz con múltiples opciones y un aceptable completamiento de código (44).

Características de NetBeans (45):

- Formado por un IDE de código abierto y una plataforma de aplicación que permite a los desarrolladores crear con rapidez aplicaciones web.
- Ofrece documentación y recursos de formación exhaustivos.
- Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, colaboración entre varias personas.

- Es un IDE multiplataforma.
- Está desarrollado para usarse generalmente con lenguaje Java, aunque permite su uso con otros lenguajes de programación como PHP.
- Permite crear aplicaciones Web con PHP 5.

Eclipse 4.4.2 es un IDE de código abierto y multiplataforma. Es una potente y completa plataforma de programación, desarrollo y compilación de elementos tan variados como sitios web, programas en C++ o aplicaciones Java así como para PHP u otros, donde se encuentran todas las herramientas y funciones necesarias para trabajar, recogidas además en una atractiva interfaz que lo hace fácil y agradable de usar (46).

Características de Eclipse (46):

- Su principal inconveniente es el consumo de recursos del sistema, que es común a otros IDE en mayor o menor medida.
- Es totalmente libre y no es necesario comprar ninguna licencia comercial, lo cual lo hace atractivo para muchos.
- Es multiplataforma.
- Es un potente editor de texto con la capacidad de resaltar sintaxis, así como también una compilación en tiempo real y soportes.

PhpStorm 8.0 es un IDE de programación desarrollado por JetBrains. Es uno de los entornos de programación más completos de la actualidad, permite editar código no sólo del lenguaje de programación PHP, como lo indica su nombre. Su editor de PHP entiende perfectamente su estructura y soporta las versiones 5.3, 5.4, 5.5 y 5.6. Entre otras características, este IDE proporciona (47):

- Finalización inteligente de código.
- Comprobación de errores al instante o mezcla de lenguajes.
- Sintaxis abreviada.

- Permite abrir un fichero individualmente y editarlo.
- Ejecución del código en la misma ventana del IDE.
- Interpretación y visualización inmediata de código PHP hasta en 5 de los navegadores web más populares. Con esto no es necesario cargar manualmente el navegador e ingresar la dirección de la página.
- Actualmente es compatible con Sistemas Operativos Windows, Linux y Mac OS.
- Con PhpStorm podemos crear nuevos proyectos basándonos en algún *framework* de diseño web y CMS (Sistema de Gestión de Contenidos, *Content Management System* por sus siglas en inglés). Como los que se muestran a continuación:
    1. Twitter Bootstrap.
    2. Symfony.
    3. HTML Boilerplate.
    4. Composer Project.
    5. Drupal.
    6. WordPress.
    7. App Engine Project.
    8. Foundation.

Fundamentos de la selección:

El autor decide utilizar PhpStorm v.8 pues soporta las tecnologías web más utilizadas actualmente. Proporciona autocompletado de código, refactorización, prevención de errores. Presenta un editor de código PHP inteligente con terminación de codificación más rápida. Permite crear proyectos utilizando *frameworks* como Bootstrap y Symfony. En lo personal, el autor cuenta con experiencia en el desarrollo web haciendo uso de esta herramienta.

1.3.5 Sistema Gestor de Base de Datos.

Un **SGBD** (Sistema Gestor de Bases de Datos) es un sistema de *software* cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Permite a los usuarios definir, crear y mantener la base de datos. Además proporciona un acceso controlado a la misma y permite a varios usuarios acceder a los datos al mismo tiempo. Los SGBD proporcionan un grupo de funciones con el objetivo de garantizar la confidencialidad, la calidad, la seguridad y la integridad de los datos que contienen, así como un acceso fácil y eficiente a los mismos (48).

PostgreSQL. 9.1

El gestor de base de datos elegido para el desarrollo del módulo que se desea implementar es PostgreSQL v9.1, esta herramienta es orientada a objetos y multiplataforma. Puede aportar potencia y flexibilidad en cuanto a las restricciones, las vistas, la herencia, las reglas y los procedimientos almacenados. Posee características significativas del motor de datos, entre las que se pueden incluir las sub consultas, los valores por defecto, las restricciones a valores en los campos (*constraints*) y los disparadores (*triggers*). Funciona en todos los sistemas operativos de Linux y debido a que es una herramienta libre se puede usar, modificar y distribuir de forma gratuita (49).

PgAdmin III.

La herramienta pgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia Open Source. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usan en Linux, FreeBSD, Solaris, Mac OS X y Windows. Es capaz de gestionar versiones a partir de PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de *software* PostgreSQL como Pervasive Postgres, EnterpriseDB, Mammoth Replicator y SRA PowerGres (50).

PgAdmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar *scripts* programados, soporte para el motor de replicación Slony-I y mucho más (50).

1.3.6 Servidor web.

Apache 2.4 es el servidor web hecho por excelencia, su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. Es una muestra, al igual que el sistema operativo Linux (un Unix desarrollado inicialmente para PC), de que el trabajo voluntario y cooperativo dentro de Internet es capaz de producir aplicaciones de calidad profesional difíciles de igualar (51). La licencia Apache es una descendiente de la licencia BSD (Licencia de *software* libre permisiva, *Berkeley Software Distribution* por sus siglas en inglés), no es GPL. Esta licencia te permite hacer lo que quieras con el código fuente (incluso *forks* y productos propietarios) siempre que les reconozcas su trabajo.

Características:

- Corre en una multitud de sistemas operativos, lo que lo hace prácticamente universal.
- Apache es una tecnología gratuita de código fuente abierta. El hecho de ser gratuito es importante pero no tanto como que se trate de código fuente abierto. Esto le da una transparencia a este *software* de manera que si se desea ver que es lo que se está instalando como servidor, se pueda saber.
- Apache es un servidor altamente configurable, de diseño modular. Es muy sencillo ampliar las capacidades del servidor web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este, y están ahí para ser instalados cuando sea necesario. Otro punto importante es que cualquier persona que posea experiencia en la programación de C o Perl puede escribir un módulo para realizar una función determinada.
- Apache te permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado *script* cuando ocurra un error en concreto.
- Tiene una alta configurabilidad en la creación y gestión de *logs*. Apache permite la creación de ficheros *log* a medida del administrador, de este modo puedes tener un mayor control sobre lo que sucede en tu servidor (51).

1.3.7 Lenguaje de modelado

UML (Lenguaje Unificado de Modelado*,* por sus siglas en inglés *Unified Modeling Language)* es un lenguaje de modelado visual para sistemas, principalmente sistemas de *software* orientados a objetos. Hoy en día, está considerado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo. Mediante UML es posible establecer una serie de requerimientos y estructuras necesarias para plasmar un sistema de *software* previo al proceso intensivo de escribir código (52).

Aunque UML es un lenguaje, posee más características visuales que programáticas, lo que facilita la fácil participación e intercomunicación de los integrantes de un equipo multidisciplinario, dígase: analistas, diseñadores, especialistas de área y desde luego los programadores. Para la realización del presente trabajo se utilizará UML.

1.3.8 Herramienta CASE de modelado.

Las herramientas CASE *(*Ingeniería de *Software* Asistida por Computadora*,* por sus siglas en inglés *Computer Aided Software Engineering)* constituyen sistemas para el modelado de *software*, que permiten aumentar la productividad reduciendo el coste de los mismos en términos de tiempo y de dinero, debido a que permiten realizar diseños del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores, entre otras funcionalidades (53).

Visual Paradigm para UML v8.0 es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. Presenta licencia gratuita y comercial. Es fácil de instalar y actualizar (54).

Visual Paradigm para UML permite visualizar, diseñar y documentar diagramas de UML 2.1 en un entorno de diseño intuitivo, ayuda al equipo de desarrollo a controlar el progreso del proyecto y brinda un medio de comunicación, facilita el modelado de un sistema donde cada involucrado puede crear sus propias vistas arquitectónicas (Vista Lógica, Vista de Procesos, Vista de Desarrollo, Vista Física y Vista de Casos de Uso) (54). Ante cambios que surjan en un negocio, permite adaptar rápidamente los modelos realizados a dichos cambios, lo cual evita escribir código sin analizar los cambios en el modelo. Es posible, descomponer un

modelado de un sistema en unidades controladas, generar código en un lenguaje específico a partir de los diseños realizados.

Conclusiones parciales.

- Los SGI estudiados no se ajustan a los requerimientos propuestos, y por tanto no satisfacen las necesidades del cliente por lo que se llega a la conclusión de que es necesario conformar una nueva solución.
- Como guía para el desarrollo del sistema se empleará la metodología Scrum.
- La herramienta CASE Visual Paradigm para UML v8.0 se utilizó para la descripción y modelación del sistema.
- Como sistema gestor de base de datos se utilizará el PostgreSQL v9.1.
- La implementación del sistema se hará empleando el IDE PhpStorm v8.0, como marcos de trabajo Symfony v2.3 del lado del servidor y Bootstrap v3 del lado del cliente.
- Como servidor Web Apache v2.4.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA. 2

Para la construcción y desarrollo de un sistema informático es de suma importancia la utilización de métodos y técnicas que permitan la solución de problemas presentes a lo largo del ciclo de vida de un *software*. La correcta captura de requisitos y el modelamiento del sistema permiten que se mitiguen las fallas que puedan aparecer durante su desarrollo, además de establecer un entendimiento común entre lo que se desea y lo que se elabora. La implementación de la solución propuesta se caracteriza por basarse en los principios y reglas de la metodología Scrum, utilizando las tecnologías y herramientas definidas.

La metodología empleada se divide en fases y actividades dentro de estas, generándose un grupo de artefactos que permiten especificar los elementos fundamentales del sistema.

2.1 Descripción del Problema.

La Muestra Joven ICAIC es un evento audiovisual que se realiza anualmente desde octubre del año 2000, reúne a jóvenes realizadores, guionistas, fotógrafos, productores, editores, sonidistas, diseñadores, entre otros especialistas. La información perteneciente a las pasadas ediciones de la Muestra Joven ICAIC se encuentra dispersa y en varios formatos, lo que hace el proceso de consulta engorroso, sin contar que la videoteca, con que cuenta la oficina de la muestra, no tiene diseñado un índice topográfico correctamente estructurado, que permita buscar una película por su título, género, director, guionista, entre otros datos.

2.2 Solución propuesta.

Se propone la implementación de un sistema que automatice el proceso de gestión de información, en la oficina de la Muestra Joven ICAIC, que facilite, el análisis de los datos de la Muestra Joven ICAIC, la evolución de sus participantes, la sistematización de sus memorias; y permita el registro de una codificación que identifique las películas de la videoteca.

2.3 Planificación del *Backlog*.

Es la fase inicial del producto donde se discute el negocio con el objetivo de tomar decisiones que impacten positivamente al producto y se especifica el propósito del proyecto. Se determina el alcance. Se comienza la creación del *Backlog* del producto para que el *sprint* siguiente contenga elementos suficientes que faciliten el trabajo (33).

2.3.1 Captura de requisitos.

La ingeniería de requisitos es el conjunto de actividades implicadas en descubrir, documentar y mantener un conjunto de requisitos (55). La captura de los mismos es un proceso en el cual los datos son extraídos de las personas pudiendo variar, dependiendo de la persona consultada.

Para la obtención de estos requisitos se emplearon técnicas que permitieron hacer este proceso de una forma más fácil y dinámica. Entre las técnicas existentes para la captura de requerimientos se empleó la entrevista, específicamente la discusión, donde se sostuvo una discusión con el cliente sobre su problemática para tratar de determinar en conjunto los requisitos del sistema.

Como resultado de la aplicación de estas técnicas se obtuvo la plantilla de Pila de productos, descrita en el siguiente epígrafe.

2.3.1.1 *Product Backlog.*

Es el inventario en el que se almacenan todas las funcionalidades o requisitos en forma de lista priorizada. Estos requisitos son los que tiene el producto y los cuales pueden incrementar en sucesivas iteraciones (33).

La Pila de Producto contiene los siguientes campos (56):

- ✓ ID: Identificador único, simplemente un número auto-incremental.
- ✓ Nombre: Descripción breve del requisito, normalmente de 2 a 10 palabras.
- ✓ Importancia: *Ratio* de importancia que el cliente da al requisito. Por ejemplo, 10. ó 150. Más alto = más importante.

- ✓ Estimación inicial: Valoración inicial del equipo referente a cuánto trabajo es necesario para implementar la funcionalidad, comparada con otras funcionalidades. La unidad son "puntos" y usualmente corresponde a "días x persona ideales". Lo importante no es que las estimaciones absolutas sean correctas (es decir, que un requisito de 2puntos deba durar 2 días), sino que las estimaciones relativas sean correctas (es decir, que un requisito de 2 puntos debería durar la mitad que una funcionalidad de 4 puntos).
- ✓ Descripción: Especificación de cómo funciona el requerimiento.

Tabla 1 *Product Backlog.*

	ID	Nombre	Importancia (0 - 100)	Estimación	Descripción
Funcionales					
	RF1	Gestionar Persona.	95	17	Insertar, modificar, listar, eliminar datos de todos los participantes de la Muestra Joven ICAIC.
	RF2	Gestionar Muestra Joven.	95	17	Insertar, modificar, listar, eliminar datos relacionados con la Muestra Joven ICAIC. Los datos que se manejan son: ✓ Número de la muestra. ✓ Fecha en que se realizó la muestra. ✓ Trípticos, posters, periódicos y catálogos en formato pdf. ✓ Jurados. ✓ Patrocinadores.
	RF3	Gestionar Película.	95	17	Insertar, modificar, listar, eliminar datos relacionados con las películas de las distintas secciones. Los datos asociados a las películas son: ✓ Código que representa su ubicación en la videoteca.

					✓ Título. ✓ Año de exhibición. ✓ Nacionalidad. ✓ Reparto.
	RF4	Gestionar Carteles.	85	11	Insertar, modificar, listar, eliminar datos relacionados con los carteles en concurso. Los datos relacionados a los carteles son: ✓ Nombre de la película. ✓ Diseñador.
	RF5	Gestionar Proyecto.	80	9	Insertar, modificar, listar, eliminar datos relacionados con los proyectos participantes en la sección Haciendo Cine. Los datos almacenados son: ✓ Nombre. ✓ Director. ✓ Productor. ✓ Guionista.
	RF6	Autenticar Usuario.	70	8	El usuario debe autenticarse para comprobar permisos de acceso.
	RF7	Gestionar Eventos Colaterales.	50	7	Insertar, modificar, listar, eliminar datos relacionados con los eventos colaterales (talleres, conferencias, homenajes). Los datos gestionados son: ✓ Tipo de evento. ✓ Descripción. ✓ Personas.
	RF8	Gestionar Exposiciones.	40	6	Insertar, modificar, listar, eliminar datos relacionados con las exposiciones realizadas en cada muestra. Los datos relacionados a estas exposiciones son: ✓ Nombre.

					✓ Lugar. ✓ Tipo de exposición.
	RF9	Cargar archivos.	20	2	Se cargan los archivos (.jpeg, .png, .pdf) relacionados a la muestra y al concurso de carteles.
	RF10	Exportar listados.	15	2	Se exportan los listados en formato .xls.
	RF11	Buscar contenido.		2	✓ Buscar contenido en todas las tablas de la base de datos mostrando la incidencia de la búsqueda en cada una de ellas. ✓ Buscar resultados en una tabla específica mediante filtros.
No Funcionales					
Usabilidad					
	RNF1	El tiempo de entrenamiento requerido para que usuarios normales y avanzados sean productivos operando el sistema es de 2 días.			
	RNF2	Debe poseer una interfaz agradable para el cliente.			
Fiabilidad					

	RNF3	El sistema estará disponible 24 horas del día, los siete días de la semana.			
	RNF4	La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores.			
Eficiencia					
	RNF5	Tiempo de respuesta promedio de las peticiones que se realizan al servidor no deberá ser mayor de 30 segundos.			
	RNF6	El número de clientes o transacciones que el sistema puede alojar es de 2000.			
Seguridad					
	RNF7	Protección contra acciones			

		no autorizadas o que puedan afectar la integridad de los datos.			
	RNF8	El sistema debe mantener en todo momento la seguridad de la información, asegurando la autenticidad de la misma.			
	RNF9	El sistema debe garantizar la confidencialidad, integridad y disponibilidad de la información que se procese en el sistema.			
	RNF10	El control de acceso se establecerá por roles que se le asignarán a los usuarios que interactúen con el sistema.			
Soporte					
	RNF11	Soporte para grandes volúmenes de datos y			

		velocidad de procesamiento.			
	RNF12	Tiempo de respuesta rápido en accesos concurrentes.			
Restricciones de diseño					
	RNF13	El lenguaje de programación es PHP 5.3.			
	RNF14	El marco de trabajo de desarrollo es Symfony 2.3, Bootstrap 3.			
	RNF15	La herramienta IDE de desarrollo utilizada será phpStorm 8.			
	RNF16	La herramienta CASE utilizada es Visual Paradigm 8.0 para el modelado.			
	RNF17	La herramienta gestora de base de datos es PostgreSQL 9.1.			
	RNF18	Servidor web Apache 2.4.			
Interfaz					

	RNF19	El sistema tiene que ofrecer una interfaz amigable, fácil de operar.			
	RNF20	Los colores a utilizar deben ser rojo, gris, blanco y negro.			
	RNF21	El *software* debe tener el nombre del evento (¨Muestra Joven ICAIC¨) y su logo.			
Software					
	RNF22	Sistema Operativo (SO) en el servidor: GNU/Linux Ubuntu 12.04 o similares, por recomendación.			
	RNF23	SO en las máquinas clientes: cualquiera que decida el cliente.			
Hardware					
	RNF24	Se recomienda para el funcionamiento mínimo			

		máquinas clientes con procesador Pentium II o versiones superiores con no menos de 256 MB (*Megabyte*) de RAM (Memoria de Acceso Aleatorio, en inglés: *Random-Access Memory*).			
	RNF25	Se recomienda para el funcionamiento óptimo máquinas clientes con procesadores Pentium IV y 512 MB de RAM.			
	RNF26	El procesador para el funcionamiento óptimo del sistema en el servidor no debe ser menor a 2 GB (*Gigabyte*) de RAM.			

2.3.1.2 *Sprint Backlog.*

Es la lista de tareas que elabora el equipo durante la planificación de un *sprint*. Se asignan las tareas a cada persona y el tiempo que queda para terminarlas. De esta forma el proyecto queda descompuesto en unidades más pequeñas pudiendo determinarse en que tareas se ha avanzado. Pueden existir dependencias entre tareas, es importante diferenciarlas. Todas las tareas deben tener un coste semejante entre 4 y 16 horas (33).

Tabla 2 *Sprint* 0.

Sprint 0				
ID	**Tarea**	**Persona**	**Estado**	**Tiempo (horas)**
T1	Generación de la base de datos.	Yohan González Espinosa	Resuelta	4
T2	Montaje del ambiente de desarrollo con la integración de los marcos de trabajo seleccionados para el desarrollo.	Yohan González Espinosa	Resuelta	4
T3	Realizar el mapeo de la base de datos.	Yohan González Espinosa	Resuelta	4
RF1T4	Implementar funcionalidad Insertar Persona.	Yohan González Espinosa	Resuelta	30
RF1T5	Implementar funcionalidad Modificar Persona.	Yohan González Espinosa	Resuelta	30
RF1T6	Implementar funcionalidad Listar Persona.	Yohan González Espinosa	Resuelta	30
RF1T7	Implementar funcionalidad Eliminar Persona.	Yohan González Espinosa	Resuelta	30
RF1T8	Diseñar las interfaces necesarias para Gestionar Persona.	Yohan González Espinosa	Resuelta	40

El listado de la planificación de las tareas de los restantes *sprint* se describe en el Anexo 2.

2.4 Arquitectura de la solución.

La arquitectura de *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. La vista

arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones (57).

Para el desarrollo del sistema se ha definido una arquitectura base para orientar el diseño de las capas lógicas a través de una propuesta de diseño; brindar una estructura física para soportar el código, creando así un esqueleto base; y proponer mecanismos de colaboración entre los componentes integrados en ella.

2.4.1 Arquitectura en 3 capas.

En el diseño de sistemas informáticos, en la actualidad, se emplean con frecuencia las arquitecturas multinivel, donde a cada capa o nivel, se le asigna una responsabilidad específica, dando lugar a que un cambio en una de ellas no influya directamente en la otra.

Capas o Niveles (58)**.**

Capa de presentación: con la cual interactúa el usuario, comunica y captura la información del usuario dando un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio.

Capa de negocio: es donde residen los programas que se ejecutan, recibiendo las peticiones del usuario y enviando las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) pues es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al sistema administrador de base de datos para almacenar o recuperar datos.

Capa de datos: es donde residen los datos. Está formada por uno o más sistemas administradores de bases de datos que realizan todo el almacenamiento, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Como especialización de la arquitectura en 3 niveles se hace uso del patrón arquitectónico Modelo – Vista – Controlador.

2.4.2 Patrón arquitectónico Modelo – Vista – Controlador.

El patrón Modelo-Vista-Controlador (MVC) es usado principalmente en entornos web como patrón por excelencia y en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se

requiere una separación de conceptos para que el desarrollo esté estructurado de una mejor manera, facilitando la programación en diferentes capas de manera paralela e independiente. MVC sugiere la separación del *software* en 3 estratos: modelo, vista y controlador, que serán explicados brevemente (59):

Modelo: Es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en contexto del sistema proveen de información al usuario.

Vista: Es la representación del modelo en forma gráfica disponible para la interacción con el usuario. En el caso de una aplicación web, la "vista" es una página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones.

Controlador: Es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el modelo en caso de ser necesario.

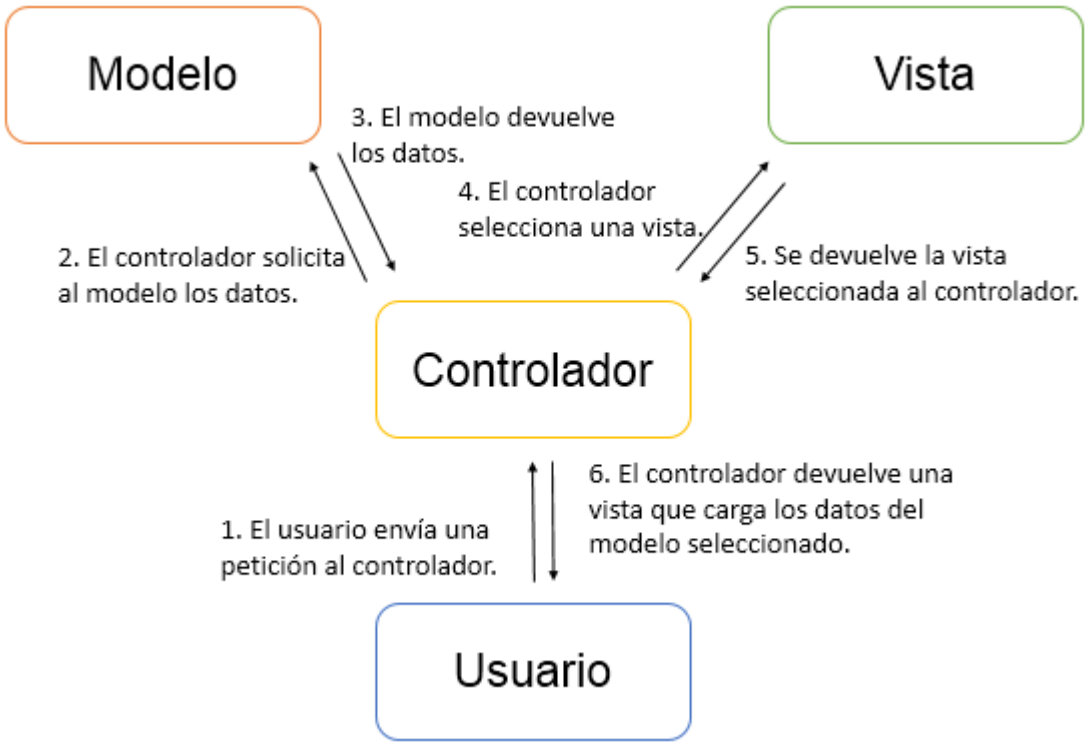

Fig. 5 Arquitectura Modelo-Vista-Controlador. Fuente (Elaboración propia).

2.5 Patrones de diseño.

La asignación de responsabilidades es la habilidad más importante en el análisis y diseño orientado por objetos, para ello tiene suma importancia la utilización de los patrones de diseño.

En términos generales, un patrón es un conjunto de información que proporciona respuesta a un conjunto de problemas similares, es decir, un patrón es una solución a un problema en un contexto (60).

2.5.1 Patrones GRASP.

"Los patrones GRASP (Patrones de *Software* para la Asignación General de Responsabilidad*, General Responsibility Assignment Software Patterns* por sus siglas en inglés) describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable." Estos patrones se describen a continuación (61):

Bajo acoplamiento*:* El Bajo acoplamiento es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. Estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento (61). Este patrón es utilizado en todas las clases del diseño.

Alta Cohesión: Asignar una responsabilidad de modo que la cohesión siga siendo alta. Representa una clase con responsabilidades moderadas en un área funcional, colaborando con otras para concretar tareas. Diseño más claro y comprensible. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme (61). Este patrón es utilizado en todas las clases del diseño.

Experto*:* Es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen. Ofrece una analogía con el mundo real (61). Este patrón se utiliza en el diseño de clases cuando se quiere decidir cuál es la clase encargada de llevar a cabo una responsabilidad con la información necesaria para realizarla.

Creador*:* Este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento, se debe buscar una clase de objeto que agregue, contenga y realice otras operaciones sobre este tipo de instancias (61).El patrón creador es utilizado en todas las clases del diseño.

Controlador*:* Es un evento generado por actores externos. Se asocian con operaciones del sistema, operaciones del sistema como respuestas a los eventos del sistema, tal como se relacionan los mensajes y los métodos (61).Este patrón se pone de manifiesto con la utilización de una clase que coordina o controla las actividades que son necesarias realizar con las demás clases, en este caso SonataCRUDControllers quien hace uso de este patrón.

2.6 Modelo de dominio.

Un modelo del dominio es una representación de las clases conceptuales del mundo real, no de componentes de *software*. Es utilizado con frecuencia para el diseño de los objetos de *software*, y será una entrada para varios de los artefactos que se verán a lo largo de la investigación. El modelo del dominio muestra clases conceptuales significativas en un dominio del problema; es el artefacto más importante que se crea durante el análisis orientado a objetos (62).

Utilizando la notación UML, un modelo del dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. Pueden mostrar (62):

- Objetos del dominio o clases conceptuales.
- Asociaciones entre las clases conceptuales.
- Atributos de las clases conceptuales.

La figura que se muestra a continuación muestra el modelo de dominio que corresponde al negocio que se ha descrito a lo largo de la presente investigación. Ilustra que, las clases conceptuales Muestra, Pelicula, Persona son significativas en este dominio, una película se encuentra relacionada a una muestra y que una persona está relacionada a la vez con una muestra y una película.

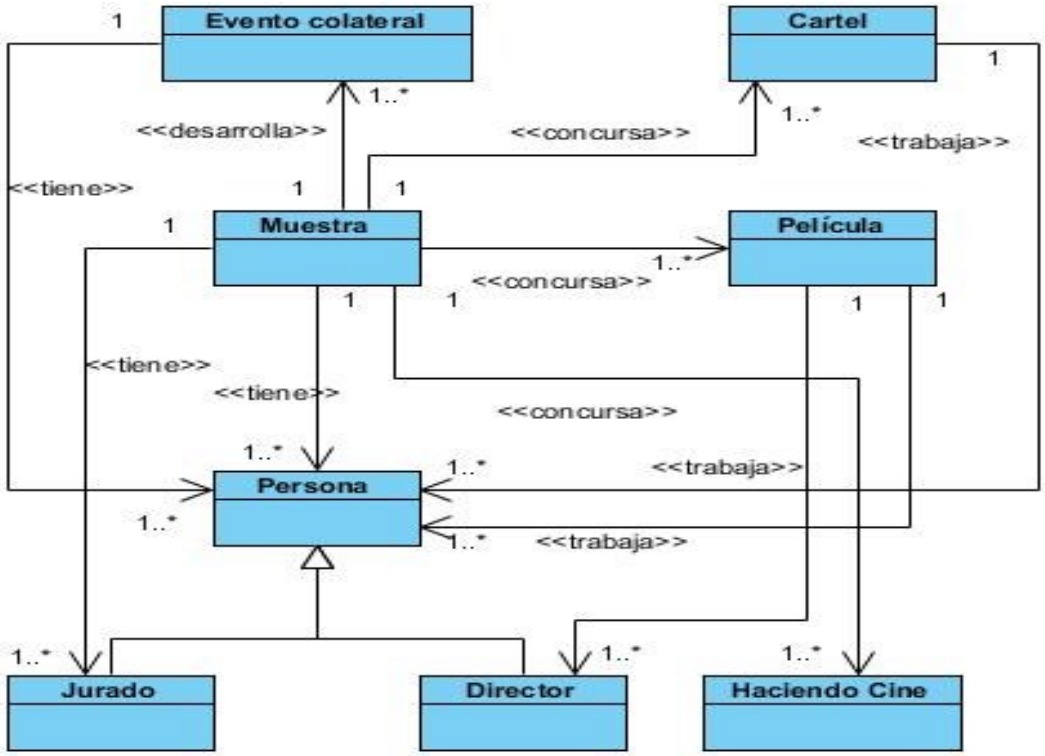

Fig. 6 Modelo de dominio.

Este diagrama fue de utilidad para lograr un entendimiento común entre el cliente y el desarrollador (autor de la investigación), pues es fácil entender los distintos elementos y sus relaciones mediante este lenguaje visual.

2.7 Modelo de datos.

Es un conjunto de conceptos que nos permiten describir los datos, las relaciones entre ellos, la semántica y las restricciones de consistencia (63).

Existen 3 tipos de modelos de datos (63):

- **Modelos externos o lógicos basados en objetos:** nos permiten representar los datos que necesita cada usuario con las estructuras propias del lenguaje de programación que se vaya a usar.
- **Modelos globales o lógicos basados en registros:** ayuda a escribir los datos para el conjunto de usuarios.

- **Modelos físicos o de datos:** orientado a la máquina.

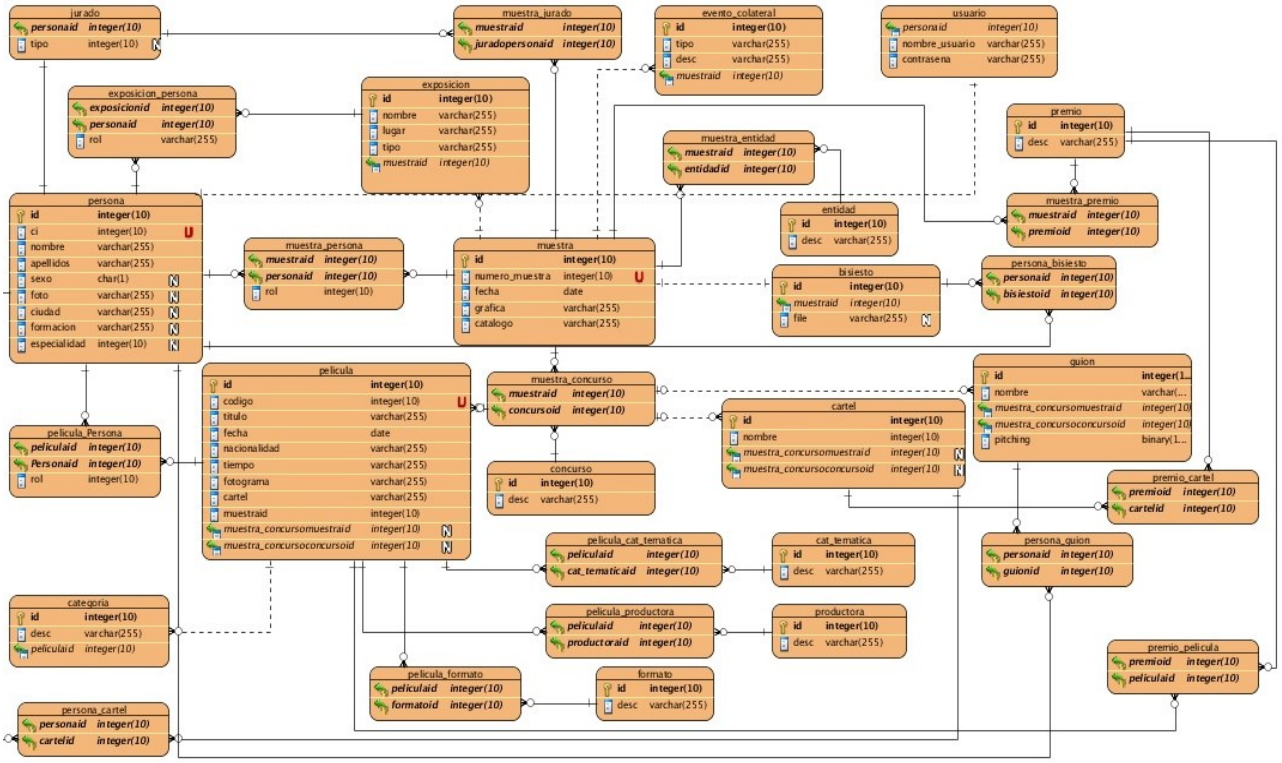

Fig. 7 Modelo de Datos.

Se describe el modelo de datos en el anexo 2, cada una de las tablas y sus atributos, un ejemplo de la descripción de estos modelos es la tabla muestra.

Tabla 3 Descripción de la tabla muestra y sus atributos.

Nombre Tabla: muestra		
Descripción: En esta tabla se almacenan los datos correspondientes a una muestra.		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador de la muestra. Es la llave primaria de la tabla.
numero_muestra	Int	Es el número de realización de la muestra.
fecha	date	Fecha de realización de la muestra.

grafica	varchar	Almacena archivos .png, .jpeg, .pdf.
catalogo	varchar	Almacena archivos en formato .pdf.

2.8 Diagrama de clases del diseño.

El diagrama de clases del diseño representa los métodos y atributos de cada una de las clases del sistema, para mostrar de forma simple la colaboración y las tareas de cada una de ellas en relación al sistema que conforman (64).

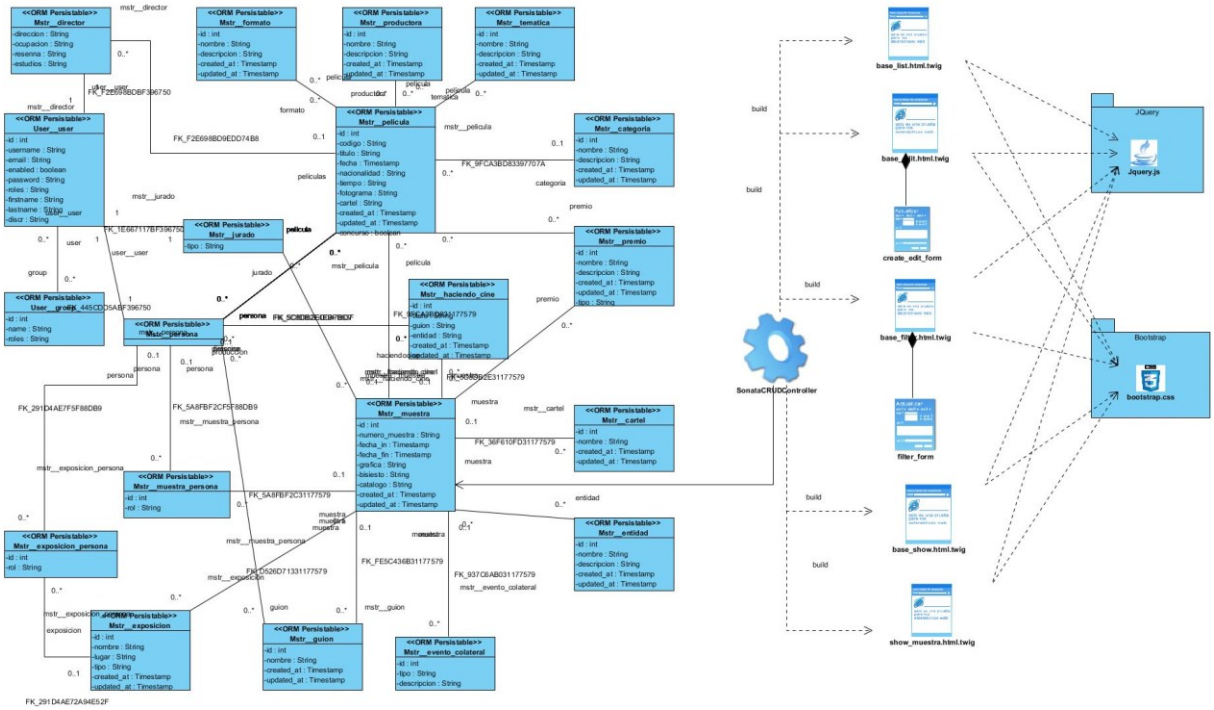

Fig. 8 Diagrama de Clases del Diseño.

En el diagrama de clases se pone de manifiesto el uso del patrón arquitectónico MVC, dada la definición de las clases interfaces que se corresponden con las vistas del patrón, ellas son base_list, base_edit, create_edit, base_filter,filter_form, base_show y show_muestra. El Controlador responde a la definición de la clase SonataCRUDController que modela el negocio implementado. El Modelo se corresponde con el diseño de 21 clases, que responden a la representación de la información que se maneja en el sistema.

Los patrones GRASP utilizados se ponen de manifiesto en el diagrama de clases, asociado a las relaciones entre las clases y las responsabilidades asociadas a cada una de ellas para dar cumplimiento a los requisitos. En el caso del patrón Creador es utilizado para dar la responsabilidad de la creación de los objetos a las clases que corresponde, el patrón Experto se utilizó para asignar las responsabilidades relacionadas con las principales funcionalidades de la aplicación. Se hizo uso del patrón Alta Cohesión, logrando que las clases tengan solo las responsabilidades que le corresponden de acuerdo a su objetivo e información que maneja.

El patrón Bajo Acoplamiento se utiliza para brindar una interfaz de comunicación entre varias clases, haciendo que las relaciones entre las clases sean optimizadas en función de hacer que el sistema sea resistente a los cambios que se puedan producir.

Cada una de las clases del diagrama se describe en el anexo 3.

Conclusiones parciales.

- Al realizarse una correcta captura de requisitos, se obtuvo una mayor claridad de las necesidades del cliente permitiendo la comprensión del sistema y posterior representación de las funcionalidades requeridas.

- Se documentaron en la Pila del Producto las principales funcionalidades a implementar, 11 requisitos funcionales y 26 no funcionales.

- Se realizó la planificación de las tareas a cumplir mediante el *Sprint Backlog*.
- Para estructurar el sistema se hace uso de del patrón arquitectónico Modelo – Vista – Controlador y para la asignación de responsabilidades se hace uso de los patrones GRAPS con el objetivo de mitigar problemas en el diseño.
- El diseño de un modelo de dominio permitió representar visualmente la información relevante del negocio en cuestión y con su presentación al cliente, un entendimiento común entre las partes.
- Con el diseño de un modelo de datos se representaron las entidades relevantes del sistema, y por medio del diagrama de clases se visualizaron las relaciones entre las clases que involucran dicho sistema, permitiendo una mejor comprensión del mismo.

- Los resultados son utilizados como entrada en la siguiente etapa de trabajo.

CAPÍTULO 3 IMPLEMENTACIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA.

3

En el presente capítulo se describen las actividades relacionadas con los procesos de implementación en términos de componentes y validación utilizando métricas de *software*. Mediante el diagrama de despliegue se detalla la distribución de la aplicación.

Por su importancia, pues con su ausencia la automatización del proceso de gestión de información en la oficina de la Muestra Joven ICAIC estaría incompleta y porque la solución planteada no gestiona contenido audiovisual, solo información, se hace necesario la creación de una codificación que represente las películas en la videoteca. Dicha codificación se detalla a continuación.

3.1 Propuesta de codificación para las películas.

Como resultado de un trabajo de mesa, en conjunto con los organizadores del evento Muestra Joven ICAIC, siendo el tema principal la importancia de la correcta gestión de la información referente a las películas, se llega al consenso de que el sistema debía recoger como atributo de la tabla pelicula un código que sirviera como identificador de la película en la videoteca. Resulta importante aclarar que en su primera versión el sistema no generará este código, solo validará que el campo siga con los parámetros que se describen a continuación:

1. El primer parámetro estará compuesto por una “M” seguida de un número, indicando el número de la muestra en la que se presentó la película.
2. El segundo parámetro se referirá a si la película está en concurso, fuera de concurso, si pertenece a la Muestra Informativa o a la sección Mirada del Otro (“C”, “FC”, “MI”, “MO” respectivamente).
3. Como tercer parámetro se tendrá en cuenta si la película es de ficción, documental, animación u otra (“F”, “D”, “A”, “O” respectivamente).
4. Por último un número consecutivo por género que no podrá repetirse.

Ejemplo: M1CA12, se refiere a la película 12 perteneciente a la primera muestra realizada, en concurso y del género animación.

Luego de insertada toda la información referente a las películas de las pasadas ediciones de la muestra, en la base de datos, la videoteca deberá ser reorganizada.

3.2 Implementación.

3.2.1 Diagrama de Componentes.

Un diagrama de componentes muestra las dependencias lógicas entre componentes de *software*, sean éstos componentes fuentes, binarios o ejecutables, ilustran las piezas del *software*, controladores embebidos, entre otros. Los diagramas de componentes prevalecen en el campo de la arquitectura de *software* pero pueden ser usados para modelar y documentar cualquier arquitectura del sistema, es decir, para describir la vista de implementación estática de un sistema. Los diagramas de componentes se relacionan con los diagramas de clases, ya que un componente normalmente se corresponde con una o más clases, interfaces o colaboraciones; pero un diagrama de componentes tiene un nivel más alto de abstracción que un diagrama de clases, usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución (65).

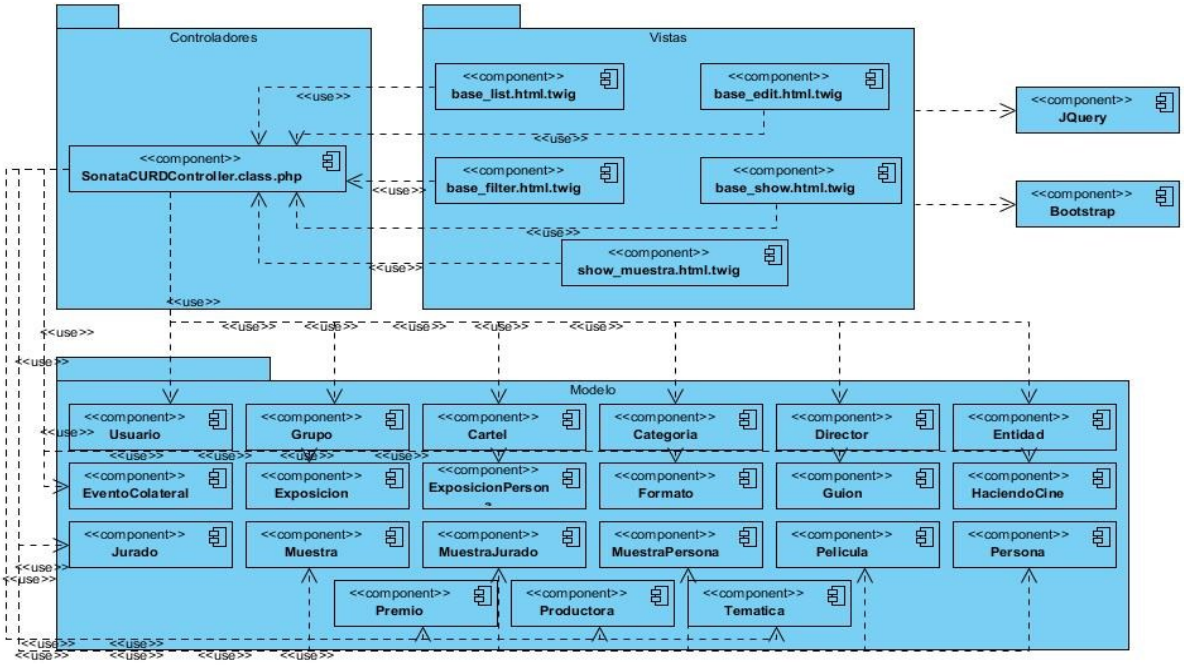

Fig. 9 Diagrama de Componentes.

3.2.2 Diagrama de Despliegue.

Un modelo de despliegue describe la arquitectura en tiempo de ejecución de un sistema. Forma parte de la vista de despliegue de UML, la cual representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. El modelo de despliegue en sí mismo representa una correspondencia entre la arquitectura de *software* y la arquitectura del sistema (el *hardware*) (66).

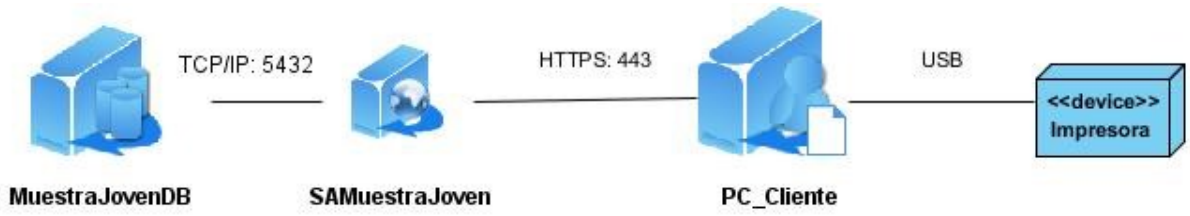

Fig. 10 Diagrama de Despliegue.

Para el despliegue del sistema a implementar se tiene que la PC_Cliente se encuentra conectada por USB a una impresora donde se podrá obtener de forma física los reportes, y por HTTPS a través del puerto 443 al servidor de aplicación. Este último mediante el protocolo TCP/IP y puerto 5432 se conectará a la base de datos del sistema.

3.3 Resultados de la aplicación de las métricas de *software* para la validación

Las métricas de *software* son una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado. Permiten averiguar cuán bien están definidas las clases y el sistema, lo cual tiene un impacto directo en el mantenimiento del mismo, tanto por la comprensión de lo desarrollado como por la dificultad de modificarlo con éxito. Estas métricas tienen como propósito entender y mejorar la calidad del producto, evaluar la efectividad del proceso y mejorar la calidad del trabajo llevado a cabo al nivel del proyecto (67).

Para la evaluación de la calidad del diseño propuesto se hará uso de las métricas Tamaño Operacional de Clase (TOC) y Relaciones entre Clases (RC) (68) (67).

TOC y RC permiten medir los siguientes atributos de calidad (69)**:**

- **Responsabilidad:** Consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- **Complejidad de implementación:** Consiste en el grado de dificultad que tiene implementado un diseño de clases determinado.
- **Reutilización:** Consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de *software*.
- **Acoplamiento:** Consiste en el grado de dependencia o interconexión de una clase o estructura de clase, con otras, está muy ligada a la característica de Reutilización.
- **Complejidad del mantenimiento:** Consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de *software*. Puede influir indirecta, pero fuertemente en los costos y la planificación del proyecto.
- **Cantidad de pruebas:** Consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad (unidad) del producto (componente, módulo, clase, conjunto de clases, entre otros.) diseñado.

3.3.1 Métrica Tamaño Operacional de Clase (TOC).

TOC está dado por el número de métodos asignados a una clase, utilizando umbrales para la responsabilidad, la complejidad y la reutilización de las clases (69).

Tabla 4 Atributos de calidad que afectan la métrica TOC (69)**.**

Atributo de Calidad	Modo en que lo afecta
Responsabilidad	Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
Complejidad de implementación	Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
Reutilización	Un aumento del TOC implica una disminución del grado de reutilización de la clase.

Tabla 5 Rango de valores para la evaluación técnica de los atributos de calidad relacionados con la métrica TOC (70).

Atributo	Categoría	Criterio
Responsabilidad	Baja	< =Promedio.
	Media	Entre Promedio y 2* Promedio
	Alta	> 2* Promedio
Complejidad de implementación	Baja	< =Promedio
	Media	Entre Promedio y 2* Promedio
	Alta	> 2* Promedio
Reutilización	Baja	> 2*Promedio
	Media	Entre Promedio y 2* Promedio
	Alta	<= Promedio.

A continuación se muestran las clases del sistema evaluadas en los atributos de calidad Responsabilidad, Complejidad de implementación y Reutilización.

Tabla 6 Evaluación de las clases del sistema mediante la métrica TOC.

Clase	Cantidad de Procedimientos	Responsabilidad	Complejidad	Reutilización
Grupo	7	Baja	Baja	Alta
Usuario	44	Alta	Alta	Baja
Cartel	10	Baja	Baja	Alta
Categoria	14	Baja	Baja	Alta
Director	2	Baja	Baja	Alta
Entidad	14	Baja	Baja	Alta
EventoColateral	8	Baja	Baja	Alta
Exposicion	18	Media	Media	Media
ExposicionPersona	7	Baja	Baja	Alta
Formato	14	Baja	Baja	Alta
Guion	15	Baja	Baja	Alta
HaciendoCine	22	Media	Media	Media
Jurado	7	Baja	Baja	Alta

Muestra	54	Alta	Alta	Baja
MuestraJurado	6	Baja	Baja	Alta
MuestraPersona	7	Baja	Baja	Alta
Pelicula	45	Alta	Alta	Baja
Persona	25	Media	Media	Media
Premio	17	Media	Media	Media
Productora	13	Baja	Baja	Alta
Tematica	14	Baja	Baja	Alta

A continuación se muestra una gráfica con agrupaciones por intervalos de la cantidad de clases según la cantidad de procedimientos:

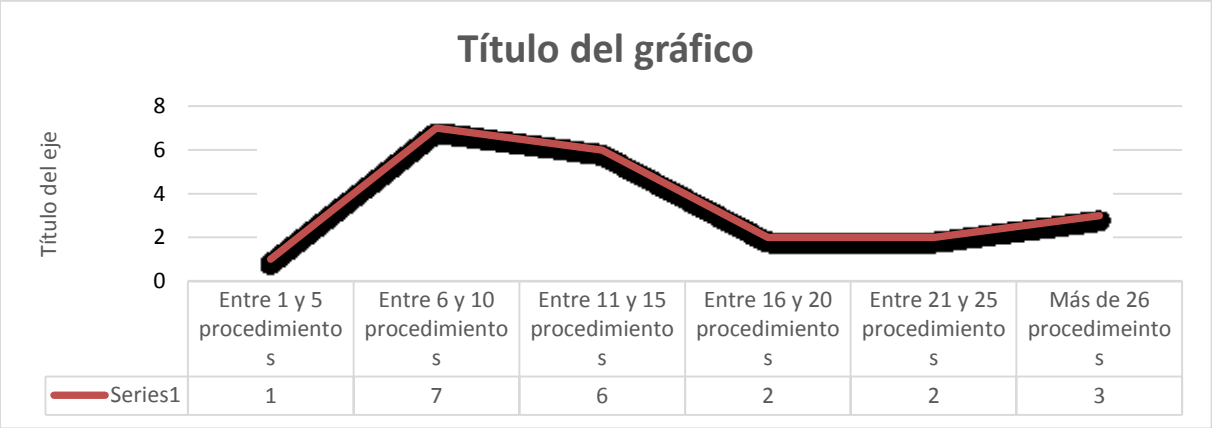

Fig. 11 Representación de la cantidad de clases agrupadas en intervalos según la cantidad de procedimientos.

En la siguiente figura se representan los resultados obtenidos al evaluar el diseño en el atributo Responsabilidad.

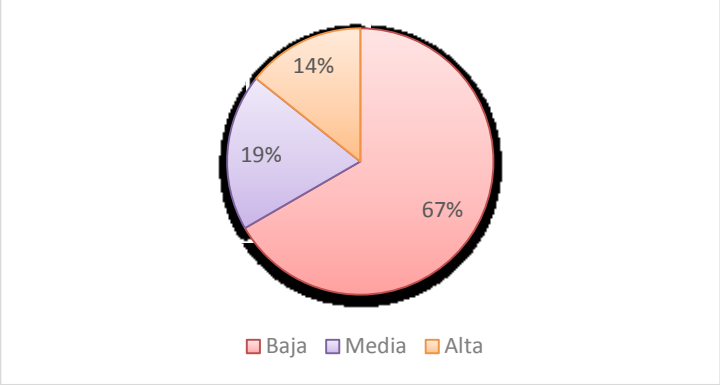

Fig. 12 Representación en porcientos (%) de los resultados obtenidos en el atributo Responsabilidad.

En la figura a continuación se muestran los resultados obtenidos luego de evaluar el atributo Complejidad de implementación.

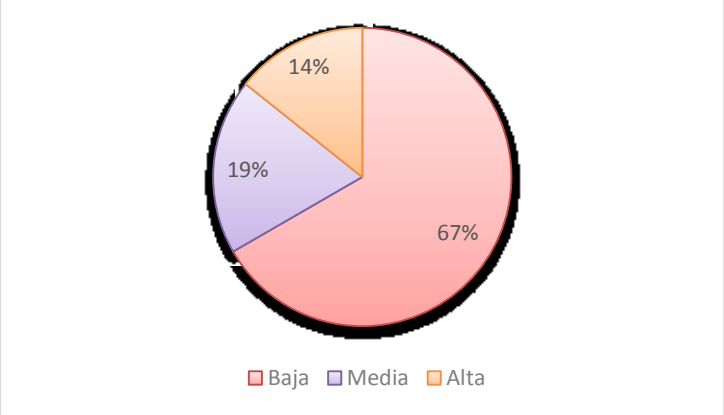

Fig. 13 Representación en porcientos (%) de los resultados obtenidos en el atributo Complejidad de implementación.

En la siguiente figura se muestran los resultados obtenidos al evaluar el atributo Reutilización.

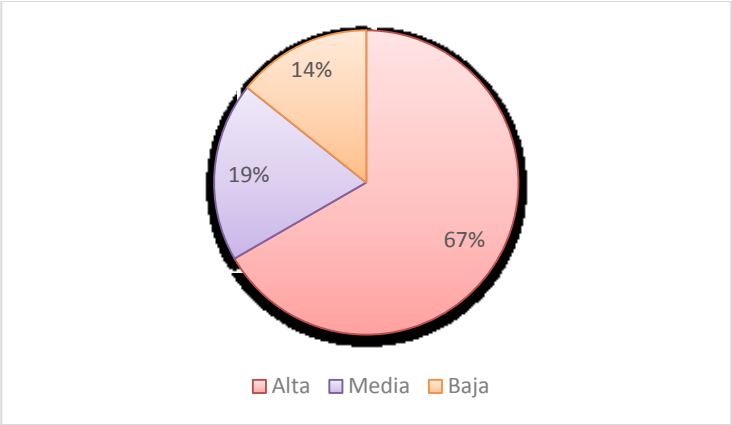

Fig. 14 Representación en porcientos (%) de los resultados obtenidos en el atributo Reutilización.

Análisis de los resultados obtenidos en la evaluación de la métrica TOC:

Durante la evaluación de la métrica TOC los resultados obtenidos demuestran que los atributos de calidad de las clases se encuentran en un nivel satisfactorio; de manera que se puede confirmar la elevada reutilización con un 67% (elemento clave en el proceso de desarrollo de *software*) y cómo se reducen la responsabilidad y la complejidad de implementación en un 67%.

3.3.1 Métrica Relaciones entre Clases (RC).

Esta métrica está dada por el número de relaciones de uso de una clase con otra. Para la evaluación de las clases fueron utilizados umbrales para el acoplamiento, complejidad de mantenimiento, la reutilización y la cantidad de pruebas (69).

Tabla 7 Atributos de Calidad para RC (69)**.**

Atributo de Calidad	Modo en que lo afecta
Acoplamiento	Un aumento del RC implica un aumento del acoplamiento de la clase.
Complejidad de mantenimiento	Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
Reutilización	Un aumento del RC implica una disminución del grado de reutilización de la clase.

Cantidad de pruebas	Un aumento del RC implica un aumento de la cantidad de pruebas de unidad necesarias para probar una clase.

Tabla 8 Rangos de valores para la evaluación técnica de los atributos de calidad relacionados con la métrica RC (70)**.**

Atributos	Categoría	Criterio
Acoplamiento	Ninguno	0
	Bajo	1
	Medio	2
	Alto	>2
Complejidad de Mantenimiento	Baja	<= Promedio
	Media	Entre Promedio. y 2* Promedio.
	Alta	> 2* Promedio.
Reutilización	Baja	>2* Promedio.
	Media	Entre Promedio. y 2* Promedio.
	Alta	<= Promedio.
Cantidad de Pruebas	Baja	<= Promedio.
	Media	Entre Promedio. y 2* Promedio.
	Alta	> 2* Promedio.

A continuación se muestran las clases del sistema evaluadas en los atributos de calidad Acoplamiento, Complejidad del mantenimiento, Cantidad de pruebas y Reutilización.

Tabla 9 Clases del sistema evaluadas en los atributos de calidad según la métrica RC.

Clase	Cantidad de relaciones de uso	Acoplamiento	Complejidad de mantenimiento	Reutilización	Cantidad de pruebas
Grupo	0	Ninguna	Baja	Alta	Baja
Usuario	0	Ninguna	Baja	Alta	Baja
Cartel	1	Baja	Baja	Alta	Baja
Categoria	1	Baja	Baja	Alta	Baja

Director	0	Ninguna	Baja	Alta	Baja
Entidad	1	Baja	Baja	Alta	Baja
EventoColateral	1	Baja	Baja	Alta	Baja
Exposicion	2	Medio	Baja	Alta	Baja
ExposicionPersona	2	Medio	Baja	Alta	Baja
Formato	1	Baja	Baja	Alta	Baja
Guion	2	Medio	Baja	Alta	Baja
HaciendoCine	4	Alto	Medio	Medio	Medio
Jurado	1	Baja	Baja	Alta	Baja
Muestra	10	Alto	Alto	Baja	Alto
MuestraJurado	2	Medio	Medio	Medio	Medio
MuestraPersona	2	Medio	Medio	Medio	Medio
Pelicula	10	Alto	Alto	Baja	Alto
Persona	6	Alto	Alto	Baja	Alto
Premio	2	Medio	Medio	Medio	Medio
Productora	1	Baja	Baja	Alta	Baja
Tematica	1	Baja	Baja	Alta	Baja

A continuación se muestra una gráfica con agrupaciones por intervalos de la cantidad de clases según las dependencias entre ellas.

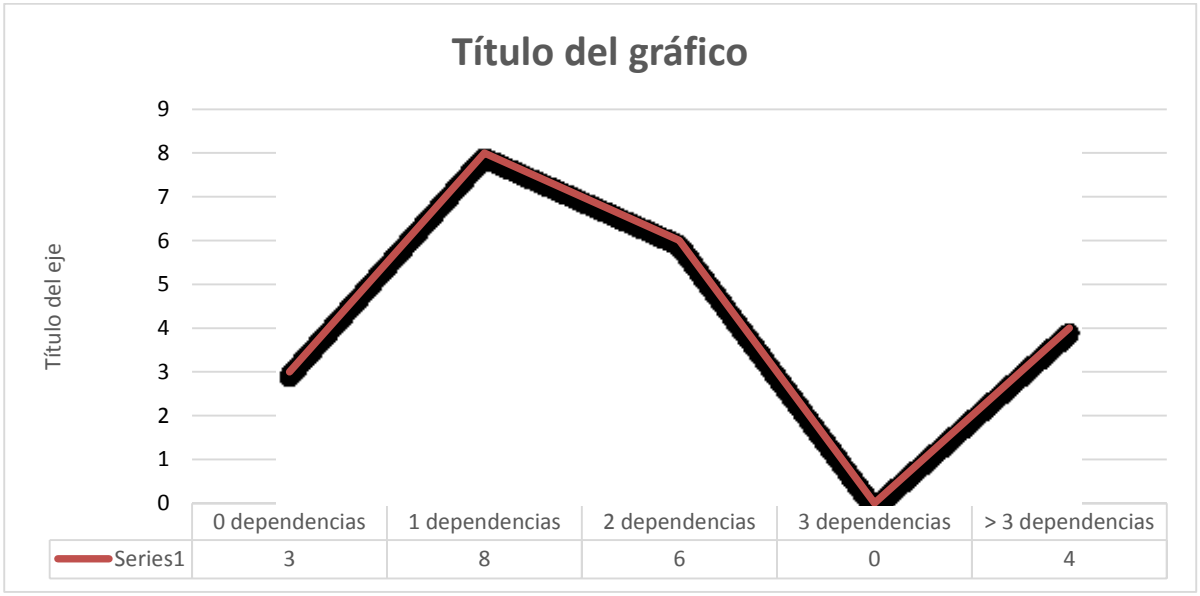

Fig. 15 Intervalos de las clases agrupadas según las dependencias entre ellas.

El sistema cuenta con 21 clases que poseen más de 3 dependencias entre ellas.

En la siguiente figura se muestran los valores, en porcientos (%), obtenidos al evaluar el diseño en el atributo Acoplamiento.

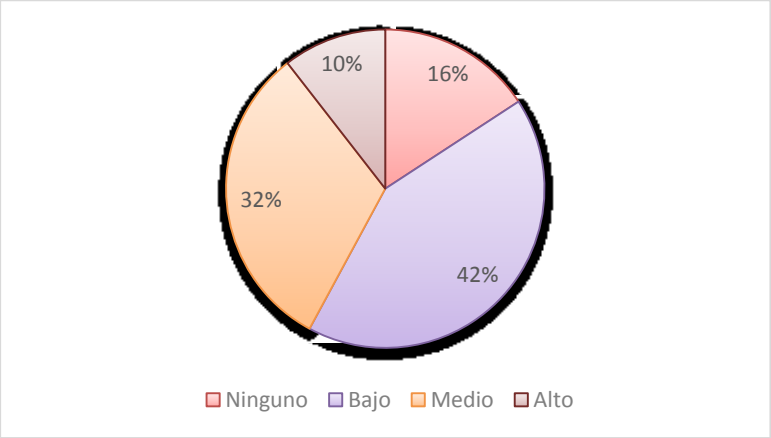

Fig. 16 Representación en porcientos (%) de los atributos obtenidos en el atributo Acoplamiento.

En la figura, a continuación, se muestran los resultados obtenidos luego de evaluar el atributo Complejidad de mantenimiento.

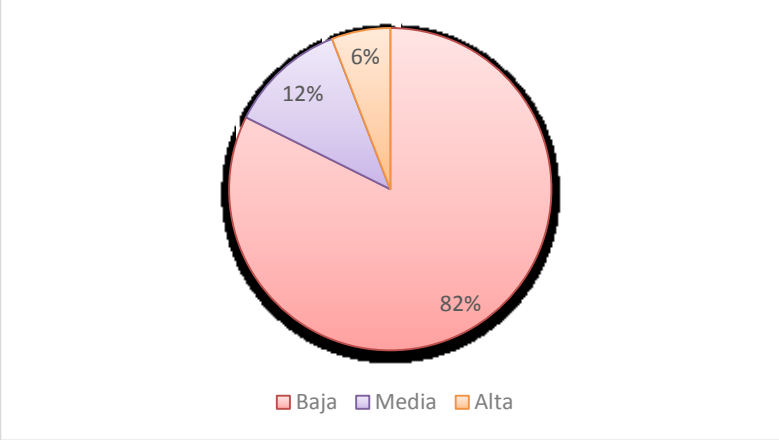

Fig. 17 Representación en porcientos (%) de los atributos obtenidos en el atributo Complejidad de mantenimiento.

La próxima figura muestra los resultados obtenidos al evaluar el atributo Cantidad de pruebas.

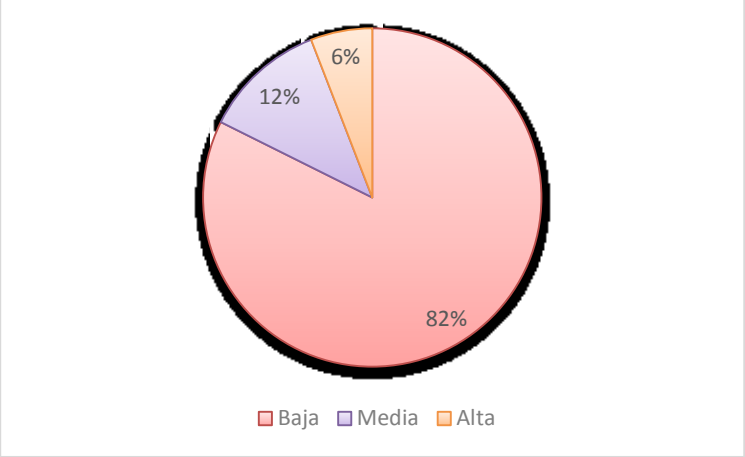

Fig. 18 Representación en porcientos (%) de los atributos obtenidos en el atributo Cantidad de pruebas.

A continuación se muestran los resultados obtenidos al evaluar el atributo Reutilización.

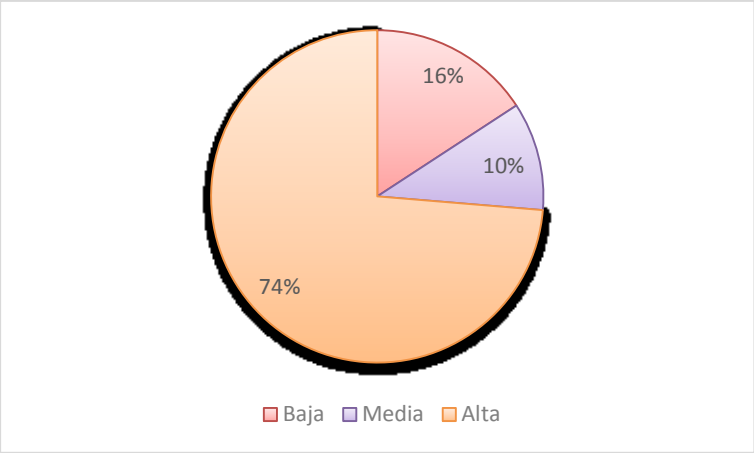

Fig. 19 Representación en porcientos (%) de los atributos obtenidos en el atributo Reutilización.

Análisis de los resultados obtenidos en la evaluación de la métrica RC:

Luego de aplicarse la métrica de diseño RC y obtenidos los resultados de la evaluación del instrumento de medición de la métrica, se puede concluir que el diseño propuesto tiene una calidad aceptable, teniendo en cuenta que el 81% de las clases empleadas poseen menos de 3 dependencias de otras clases; lo que conlleva a evaluaciones positivas de los atributos de calidad involucrados (bajo acoplamiento con 42%, baja complejidad de mantenimiento con un 82%, baja cantidad de pruebas con 82% y alta reutilización con un

74%); favoreciendo de esta manera la reutilización de las clases, así como la modificación e implantación del diseño.

Conclusiones parciales.

Como resultado de este capítulo:

- Se propuso una codificación que identificará las películas en la videoteca para de esta forma facilitar su búsqueda en el sistema por director, guionista, entre otros datos.
- Se realizó el diagrama de componentes y de despliegue, mostrando una vista de la aplicación a nivel de componentes y su distribución.
- Se aplicaron las métricas TOC y RC que arrojaron resultados positivos en la valoración de la validación de la aplicación propuesta.
- Con las validaciones realizadas se puede concluir que la herramienta desarrollada cumple con las especificaciones y requisitos definidos por los clientes en la etapa de concepción del sistema.

CONCLUSIONES

Durante el desarrollo de la presente investigación, descrita en el documento, se ha dado cumplimiento a los objetivos planteados, construyendo un sistema automatizado de gestión de información, para la Muestra Joven ICAIC, que permite el análisis de los datos, la evolución de los participantes, la sistematización de las memorias y por tanto el proceso de recuperación de información de las ediciones del evento.

Entre las principales conclusiones a las que se puede arribar con la investigación y trabajo desarrollado se encuentran:

- Se realizó un estudio de los SGI arrojando como resultado que los mismos no se ajustan a los requerimientos propuestos, y por tanto no satisfacen las necesidades del cliente por lo que se conformó una nueva solución.
- El estudio correspondiente a las herramientas y tecnologías permitió que su elección fuese la más adecuada para. la correcta y rápida implementación del sistema.
- Se ejecutaron las etapas y actividades correspondientes a la metodología definida, generándose los artefactos correspondientes permitiendo una mejor comprensión y documentación del sistema implementado con vistas a futuros mantenimientos y escalabilidad de la aplicación.
- Se realizó la validación del sistema implementado mediante métricas de calidad. Esta validación arrojó como resultados que la aplicación desarrollada satisface los requerimientos, así como el diseño de las clases e implementación de acuerdo a las métricas aplicadas.

RECOMENDACIONES

Para una futura versión del sistema se recomienda:

- ✓ Desarrollar la generación automática de la codificación que tributan a las películas en la videoteca.
- ✓ Implementar un módulo de gestión audiovisual que permita el almacenamiento de las películas en sus diferentes formatos.

BIBLIOGRAFÍA REFERENCIADA

1. **Ponjuán, G.** El profesional de la informción y su dominio de las técnicas y herramientas de la Gestión. *Aplicaciones de Gestión de información en las organizaciones.* La Habana : s.n., 2000.

2. **Muestra Joven.** Muestra Joven ICAIC. [En línea] http://muestrajoven.cult.cu/.

3. **Métodos teóricos. [En línea] http://www.buenastareas.com/ensayos/Metodos-Teoricos/136411.html.**

4. Zayas, A.P.M. ***EL ROMBO DE LAS INVESTIGACIONES DE LAS CIENCIAS SOCIALES.***

5. Métodos Empíricos. [En línea] http://es.scribd.com/doc/21229743/METODOS-EMPIRICOS.

6. OBSERVACIÓN DESCRIPTIVA Y EXPERIMENTO. [En línea] HTTP://WWW2.UIAH.FI/PROJECTS/METODI/262.HTM.

7. Custodio, R.A. MÉTODOS Y TÉCNICAS. [En línea] 2008. Custodio, R.A. MÉTODOS Y TÉCNICAS. http://www.gestiopolis.com/economia/tecnicas-y-metodos-de-investigacion.htm..

8. M., Peralta. "Sistema de Información". [En línea] 8 de Septiembre de 2006. http://www.monografias.com/trabajos7/sisinf/sisinf.shtml.

9. Bustelo Ruesta C, Garcia Morales Huidobro E. La consultoria en organizaciones. ***El profesional de la información.*** **200.**

10. N, Angulo Marcial. ***Información: una nueva propuesta conceptual.*** **s.l. : Cienc Inform, 1996.**

11. ***IRM Concepts: Building blocks for the 1990´s.*** **Owen, D.E. 2, s.l. : Information Management Review, 1989, Vol. V.**

12. Paez Urdaneta, I. ***Gestión de la inteligencia, aprendizaje tecnológico y modernización del trabajo intelectual: retos y oportunidades.*** **Caracas : s.n., 1998.**

13. Ponjuán Dante, G. ***Gestión de Información en la Organizaciones. Centro de CECAPI.*** **Chile : Universidad de Chile, 1998.**

14. Hernández, Vega Brian. ***DISEÑO DE PROTOTIPO DE UN SISTEMA DE GESTIÓN DE LA INFORMACIÓN PARA El GRUPO DE TRABAJO DE ALIMENTOS Y BEBIDAS DEL HOTEL"BRISAS SANTA LUCIA" .*** **2007.**

15. ***Análisis evolutivo de los sistemas de información y su marco conceptual.*** **2013.**

16. SIS, Comisión de Carrera. ***Programa de la Asignatura Seguridad, ética y auditoría informática. Facultad de Tecnología de la Salud "Salvador Allende". Universidad de Ciencias Médicas de La Habana.*** **La Habana : s.n., 2006.**

17. O'BRIEN, James A. ***Sistemas de Información Gerencial.*** **[ed.] Irwin McGraw-Hill. Cuarta. 2001. pág. 700.**

18. ***Procedimiento para implantar el sistema de gestión de la calidad en Centros de Información y Gestión Tecnológica.*** **Labrada-Pino, I. 4, Holguín : Ciencias Holguín, 2013, Vol. XIX.**

19. ***Gestión de la información y el conocimiento en las organizaciones.*** **Rangelov, Youlianov Stanislav. 12, Lima : Biblios, Junio de 2002.**

20. Vergara, Gonzalo. Mejora tu gestión. ***¿Que es un SISTEMA de GESTION?*** **[En línea] 31 de marzo de 2009. http://mejoratugestion.com/mejora-tu-gestion/que-es-un-sistema-de-gestion/.**

21. Nuñes Marcos, Rafael. Sistemas Informáticos II. ***Tecnicatura superior en seguridad pública especialización en informática.*** **2014.**

22. Distraidos . ¿App Nativas o App Webs? Cual es mejor!? . [En línea] http://www.distraidos.org/2012/06/21/app-nativas-o-app-webscual-es-mejor/.

23. Mastermagazine. Definición de Aplicación. [En línea] http://www.mastermagazine.info/termino/3874.php..

24. Pérez, Guevara. ***Apliación informática para la gestión de la información asociada al grupo de gestión de la continuidad de los servicios tecnológicos de la información.*** **Universidad de las Cienicias Informáticas. La Habana : s.n., 2013. Tesis.**

25. ZyLAB. ZyLAB. ***Discovery & Information Management Software and Services.*** **[En línea] http://www.zylab.com/.**

26. Orange HRM. AplicacionesEmpresariales.com. [En línea] 2008. http://www.aplicacionesempresariales.com/orange-hrm-control-de-personal.html.

27. ASSETS. Página oficial de ASSETS. [En línea] 2004. http://www.assets.co.cu/ventajas.asp.

28. Figueroa, R.G., Solís, Camilo J., Cabrera, Armando A. ***Metodologías tradicionales vs metodologías ágiles.*** **2011.**

29. Infante, L. ***Metodología Agil.*** **2009.**

30. Canós, J.H., Letelier, Patricio, Panadés, M. Carmen. Metodologías ágiles en el desarrollo de ***software*****. [En línea] http://www.willydev.net/descargas/prev/TodoAgil.Pdf..**

31. Pérez, María José Pérez. ***Guía Comparativa de Metodologías Ágiles.*** **Valladolid : s.n.**

32. Palacio, Juan, Ruata, Claudia. ***Scrum Manager. Gestión de Proyectos.*** **2009.**

33. Gallego, Manuel Trigas. ***Metodología Scrum.***

34. ClubDesarrolladores. Ventajas de la Metodología Scrum. [En línea] http://www.clubdesarrolladores.com/articulos/mostrar/63-metodologia-scrum/2..

35. Wilson, Leslie B. *Comparative Programming Languages.* Segunda. 1993.

36. Java. Código de Programación. [En línea] http://codigoprogramacion.com/java/47-introjava.html.

37. Mehdi Achour, F.B., Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, Jakub Vrana. PHP Manual 2012. [En línea] 2012. http://www.php.net/manual/en/intro-whatcando.php.

38. Pérez, J.E. Introducción a JavaScript. [En línea] http://www.librosweb.es/javascript/index.html.

39. Eidos, Grupo. Lenguaje Html. [En línea] 2011. http://www.matematica.ciens.ucv.ve/files/Manuales/Manuales/Programacion%20Web%20-%20Lenguaje%20HTML.pdf.

40. Pérez, Javier Eguíluz. Introducción a los CSS. [En línea] 2012. http://www.fim.umich.mx/var/cursoweb/introduccion_css_2caras.pdf.

41. López, Esteban Saavedra. *Frameworks para desarrollo de aplicaciones Web.* 2008.

42. Zaninotto, F., Potencier F. *The Definitive Guide to symfony.* 2007.

43. Otto, Mark. *Bootstrap in A List Apart #342.* 2012.

44. Domínguez-Dorado, M. *NetBeans IDE 4.1. La alternativa a Eclipse.* Madrid : Iberprensa, 2005.

45. NetBeans. Página Oficial de NetBeans. [En línea] 2012. http://netbeans.org/.

46. Language Packs: 3.2. Página Oficial de Eclipse. [En línea] eclipse.org.

47. JetBrains. Sitio oficial de JetBrains. [En línea] 2000. https://www.jetbrains.com/phpstorm/.

48. Gil, Fidel . *SISTEMAS DE GESTIÓN DE BASE DE DATOS.* 2005.

49. Lockhart, Tyler. *Manual de usuario de PostgreSQL.* 1996.

50. Ocampo, Sebastian. PgAdmin III. [En línea] 2009. http://www.guia-ubuntu.com/index.php?title=PgAdmin_III..

51. Caceres. Servidor Web Apache. [En línea] 2012. http://linux.ciberaula.com/articulo/linux_apache_intro. .

52. Osmosislatina. [En línea] Febrero de 2009. http://www.osmosislatina.com/lenguajes/uml/basico.htm.

53. Itescam. Las Herramientas CASE. [En línea] 2012. www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r19670.DOC.

54. Zapata, Carlos. Visual Paradigm for UML. [En línea] 2014. http://www.todoprogramas.com/programalinux/visualparadigmforuml/.

55. Peñalver, G.M., A. García, S. *SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE.* Chile : s.n., 2010.

56. Kniberg, Henrik. *SCRUM Y XP DESDE LAS TRINCHERAS.* 2007.

57. Clements, P. *"A Survey of Architecture Description Languages", in Proceedings of the International Workshop on Software Specification and Design.* Alemania : s.n., 1996.

58. Arquitectura de 3 niveles. *Maestría en tecnologías de información. Aplicaciones web.*

59. Catarina. Arquitectura del *Software*. [En línea] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rivera_l_a/capitulo2.pdf..

60. GRASP. Patrones Grasp. [En línea] 2011. http://www.buenastareas.com/ensayos/Patrones-Grasp/1896730.html..

61. Areces, G.A., Diaz, Marquez Iskael. Diseño e implementación de las capas de negocio y acceso a datos de los módulos Planificación y Ejecución de Visitas Familiares. 2008.

62. Larman, Craig. MODELO DEL DOMINIO. *UML y Patrones.* Segunda. s.l. : Prentice Hall, 2003.

63. Slides. Diagrama de clases. [En línea] 2005. http://www-2.dc.uba.ar/materias/isoft1/is1-2005_2/apuntes/SlidesDC.pdf..

64. Diagrama de Clases. [En línea] 2005. http://www-2.dc.uba.ar/materias/isoft1/is1-2005_2/apuntes/SlidesDC.pdf..

65. Arizaca, R.E. *Artefacto: Diagrama de Componentes.* La Paz : s.n., 2009.

66. *Ingeniería para productos con tecnología multimedia.* Díaz, S. 2007. II Taller de *Software* Educativo y Multimedia.

67. PRESSMAN, R. *Ingeniería del Software: Un Enfoque Práctico.* 2005.

68. LORENZ, M., J. KIDD. *Object-oriented software metrics Journal of Systems and Software.* 1994.

69. Driggs Vélez, J.C. *Diseño e Implementación de la nueva versión del Módulo Banco del Sistema Integral de Gestión Cedrux.* 2011.

70. Verdecia, E. *Metodología para la formación formativa de roles desde la práctica profesional.* La Habana : s.n., 2011.

71. Palacio, Juan y Ruata, Claudia. *Scrum Manager Gestión de Proyectos.* 2009.

72. Carralero, Amilcar. Ubuntu. [En línea] 2009. [Citado el: 3 de Noviembre de 2013.] http://ciudaddeguatemala.olx.com.gt/que-es-ubuntu-iid-146168031..

73. Martínez, Martha. Lenguaje de Programacion Ruby. [En línea] 2014. http://es.scribd.com/doc/28255974/Lenguaje-de-Programacion-Ruby.

74. Ramírez, Waldo. Net Beans Caracteríscas. [En línea] 2009. https://netbeans.org/community/releases/69/index_es.html..

75. Zapata, Javier. Niveles de pruebas de *software*. [En línea] 2013. http://pruebasdel*software*.wordpress.com/.

76. EcuRed. IDE de prgramación. [En línea] http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n#Ejemplos_de_IDE_de_programaci.C3.B3n.

77. —. Eclipse, entorno de desarrollo integrado. [En línea] http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado..

78. Pérez, J.E. Introducción a JavaScript. [En línea] http://www.librosweb.es/javascript/index.html.

79. Goñi Camejo, Ivis. *Algunas reflexiones sobre el concepto de información y sus implicaciones para el desarrollo de las ciencias de la información.* s.l. : ACIMED, 2000.

80. CABRERA, E. "Control". [En línea] 2005. www.monografias.com/trabajos14/control/control.shtml.

81. Adams, Elizabeth. *La gestión de recursos de información: importancia, desafíos y responsabilidades.* 2002.

82. *"Diseño e implementación de un sistema de información para el control del combustible en la empresa de suministros y transporte agropecuarios de Sancti Spíritus".* Comas Rodríguez, Nogueira Rivera, Gutiérrez Morales, Romero Bartutis. 144, 2011, Observatorio de la Economía Latinoamericana.

83. Vickery, Brian. *Information Science in Theory and Practice.* 1997.

84. Cohen, D. *Sistemas de Información para la toma de decisiones.* 1996.

85. Norma Cubana NC ISO/IEC.

86. HAX, A. y MAJLUF, N. *Estrategias para el Liderazgo Competitivo. De la visión a los resultados.* s.l. : Dolmen, 1997.

87. Centro del Conocimiento de Fundación EDE. *Guía para la evaluación y mejora de procesos de gestión.* 2011.

BIBLIOGRAFÍA CONSULTADA

1. **Ponjuán, G.** El profesional de la informción y su dominio de las técnicas y herramientas de la Gestión. *Aplicaciones de Gestión de información en las organizaciones.* La Habana : s.n., 2000.

2. **Muestra Joven.** Muestra Joven ICAIC. [En línea] http://muestrajoven.cult.cu/.

3. **Métodos teóricos. [En línea] http://www.buenastareas.com/ensayos/Metodos-Teoricos/136411.html.**

4. Zayas, A.P.M. ***EL ROMBO DE LAS INVESTIGACIONES DE LAS CIENCIAS SOCIALES.***

5. Métodos Empíricos. [En línea] http://es.scribd.com/doc/21229743/METODOS-EMPIRICOS.

6. OBSERVACIÓN DESCRIPTIVA Y EXPERIMENTO. [En línea] HTTP://WWW2.UIAH.FI/PROJECTS/METODI/262.HTM.

7. Custodio, R.A. MÉTODOS Y TÉCNICAS. [En línea] 2008. Custodio, R.A. MÉTODOS Y TÉCNICAS. http://www.gestiopolis.com/economia/tecnicas-y-metodos-de-investigacion.htm..

8. M., Peralta. "Sistema de Información". [En línea] 8 de Septiembre de 2006. http://www.monografias.com/trabajos7/sisinf/sisinf.shtml.

9. Bustelo Ruesta C, Garcia Morales Huidobro E. La consultoria en organizaciones. ***El profesional de la información.*** **200.**

10. N, Angulo Marcial. ***Información: una nueva propuesta conceptual.*** **s.l. : Cienc Inform, 1996.**

11. ***IRM Concepts: Building blocks for the 1990´s.*** **Owen, D.E. 2, s.l. : Information Management Review, 1989, Vol. V.**

12. Paez Urdaneta, I. ***Gestión de la inteligencia, aprendizaje tecnológico y modernización del trabajo intelectual: retos y oportunidades.*** **Caracas : s.n., 1998.**

13. Ponjuán Dante, G. ***Gestión de Información en la Organizaciones. Centro de CECAPI.*** **Chile : Universidad de Chile, 1998.**

14. Hernández, Vega Brian. ***DISEÑO DE PROTOTIPO DE UN SISTEMA DE GESTIÓN DE LA INFORMACIÓN PARA El GRUPO DE TRABAJO DE ALIMENTOS Y BEBIDAS DEL HOTEL"BRISAS SANTA LUCIA" .*** **2007.**

15. ***Análisis evolutivo de los sistemas de información y su marco conceptual.*** **2013.**

16. SIS, Comisión de Carrera. ***Programa de la Asignatura Seguridad, ética y auditoría informática. Facultad de Tecnología de la Salud "Salvador Allende". Universidad de Ciencias Médicas de La Habana.*** **La Habana : s.n., 2006.**

17. O'BRIEN, James A. ***Sistemas de Información Gerencial.*** **[ed.] Irwin McGraw-Hill. Cuarta. 2001. pág. 700.**

18. ***Procedimiento para implantar el sistema de gestión de la calidad en Centros de Información y Gestión Tecnológica.*** **Labrada-Pino, I. 4, Holguín : Ciencias Holguín, 2013, Vol. XIX.**

19. ***Gestión de la información y el conocimiento en las organizaciones.*** **Rangelov, Youlianov Stanislav. 12, Lima : Biblios, Junio de 2002.**

20. Vergara, Gonzalo. Mejora tu gestión. ***¿Que es un SISTEMA de GESTION?*** **[En línea] 31 de marzo de 2009. http://mejoratugestion.com/mejora-tu-gestion/que-es-un-sistema-de-gestion/.**

21. Nuñes Marcos, Rafael. Sistemas Informáticos II. ***Tecnicatura superior en seguridad pública especialización en informática.*** **2014.**

22. Distraidos . ¿App Nativas o App Webs? Cual es mejor!? . [En línea] http://www.distraidos.org/2012/06/21/app-nativas-o-app-webscual-es-mejor/.

23. Mastermagazine. Definición de Aplicación. [En línea] http://www.mastermagazine.info/termino/3874.php..

24. Pérez, Guevara. ***Apliación informática para la gestión de la información asociada al grupo de gestión de la continuidad de los servicios tecnológicos de la información.*** **Universidad de las Cienicias Informáticas. La Habana : s.n., 2013. Tesis.**

25. ZyLAB. ZyLAB. ***Discovery & Information Management Software and Services.*** **[En línea] http://www.zylab.com/.**

26. Orange HRM. AplicacionesEmpresariales.com. [En línea] 2008. http://www.aplicacionesempresariales.com/orange-hrm-control-de-personal.html.

27. ASSETS. Página oficial de ASSETS. [En línea] 2004. http://www.assets.co.cu/ventajas.asp.

28. Figueroa, R.G., Solís, Camilo J., Cabrera, Armando A. ***Metodologías tradicionales vs metodologías ágiles.*** **2011.**

29. Infante, L. ***Metodología Agil.*** **2009.**

30. Canós, J.H., Letelier, Patricio, Panadés, M. Carmen. Metodologías ágiles en el desarrollo de ***software*****. [En línea] http://www.willydev.net/descargas/prev/TodoAgil.Pdf..**

31. Pérez, María José Pérez. ***Guía Comparativa de Metodologías Ágiles.*** **Valladolid : s.n.**

32. Palacio, Juan, Ruata, Claudia. ***Scrum Manager. Gestión de Proyectos.*** **2009.**

33. Gallego, Manuel Trigas. ***Metodología Scrum.***

34. ClubDesarrolladores. Ventajas de la Metodología Scrum. [En línea] http://www.clubdesarrolladores.com/articulos/mostrar/63-metodologia-scrum/2..

35. Wilson, Leslie B. *Comparative Programming Languages.* Segunda. 1993.

36. Java. Código de Programación. [En línea] http://codigoprogramacion.com/java/47-introjava.html.

37. Mehdi Achour, F.B., Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, Jakub Vrana. PHP Manual 2012. [En línea] 2012. http://www.php.net/manual/en/intro-whatcando.php.

38. Pérez, J.E. Introducción a JavaScript. [En línea] http://www.librosweb.es/javascript/index.html.

39. Eidos, Grupo. Lenguaje Html. [En línea] 2011. http://www.matematica.ciens.ucv.ve/files/Manuales/Manuales/Programacion%20Web%20-%20Lenguaje%20HTML.pdf.

40. Pérez, Javier Eguíluz. Introducción a los CSS. [En línea] 2012. http://www.fim.umich.mx/var/cursoweb/introduccion_css_2caras.pdf.

41. López, Esteban Saavedra. *Frameworks para desarrollo de aplicaciones Web.* 2008.

42. Zaninotto, F., Potencier F. *The Definitive Guide to symfony.* 2007.

43. Otto, Mark. *Bootstrap in A List Apart #342.* 2012.

44. Domínguez-Dorado, M. *NetBeans IDE 4.1. La alternativa a Eclipse.* Madrid : Iberprensa, 2005.

45. NetBeans. Página Oficial de NetBeans. [En línea] 2012. http://netbeans.org/.

46. Language Packs: 3.2. Página Oficial de Eclipse. [En línea] eclipse.org.

47. JetBrains. Sitio oficial de JetBrains. [En línea] 2000. https://www.jetbrains.com/phpstorm/.

48. Gil, Fidel . *SISTEMAS DE GESTIÓN DE BASE DE DATOS.* 2005.

49. Lockhart, Tyler. *Manual de usuario de PostgreSQL.* 1996.

50. Ocampo, Sebastian. PgAdmin III. [En línea] 2009. http://www.guia-ubuntu.com/index.php?title=PgAdmin_III..

51. Caceres. Servidor Web Apache. [En línea] 2012. http://linux.ciberaula.com/articulo/linux_apache_intro. .

52. Osmosislatina. [En línea] Febrero de 2009. http://www.osmosislatina.com/lenguajes/uml/basico.htm.

53. Itescam. Las Herramientas CASE. [En línea] 2012. www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r19670.DOC.

54. Zapata, Carlos. Visual Paradigm for UML. [En línea] 2014. http://www.todoprogramas.com/programalinux/visualparadigmforuml/.

55. Peñalver, G.M., A. García, S. *SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE.* Chile : s.n., 2010.

56. Kniberg, Henrik. *SCRUM Y XP DESDE LAS TRINCHERAS.* 2007.

57. Clements, P. *"A Survey of Architecture Description Languages", in Proceedings of the International Workshop on Software Specification and Design.* Alemania : s.n., 1996.

58. Arquitectura de 3 niveles. *Maestría en tecnologías de información. Aplicaciones web.*

59. Catarina. Arquitectura del *Software*. [En línea] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rivera_l_a/capitulo2.pdf..

60. GRASP. Patrones Grasp. [En línea] 2011. http://www.buenastareas.com/ensayos/Patrones-Grasp/1896730.html..

61. Areces, G.A., Diaz, Marquez Iskael. Diseño e implementación de las capas de negocio y acceso a datos de los módulos Planificación y Ejecución de Visitas Familiares. 2008.

62. Larman, Craig. MODELO DEL DOMINIO. *UML y Patrones.* Segunda. s.l. : Prentice Hall, 2003.

63. Slides. Diagrama de clases. [En línea] 2005. http://www-2.dc.uba.ar/materias/isoft1/is1-2005_2/apuntes/SlidesDC.pdf..

64. Diagrama de Clases. [En línea] 2005. http://www-2.dc.uba.ar/materias/isoft1/is1-2005_2/apuntes/SlidesDC.pdf..

65. Arizaca, R.E. *Artefacto: Diagrama de Componentes.* La Paz : s.n., 2009.

66. *Ingeniería para productos con tecnología multimedia.* Díaz, S. 2007. II Taller de *Software* Educativo y Multimedia.

67. PRESSMAN, R. *Ingeniería del Software: Un Enfoque Práctico.* 2005.

68. LORENZ, M., J. KIDD. *Object-oriented software metrics Journal of Systems and Software.* 1994.

69. Driggs Vélez, J.C. *Diseño e Implementación de la nueva versión del Módulo Banco del Sistema Integral de Gestión Cedrux.* 2011.

70. Verdecia, E. *Metodología para la formación formativa de roles desde la práctica profesional.* La Habana : s.n., 2011.

71. Palacio, Juan y Ruata, Claudia. *Scrum Manager Gestión de Proyectos.* 2009.

72. Carralero, Amilcar. Ubuntu. [En línea] 2009. [Citado el: 3 de Noviembre de 2013.] http://ciudaddeguatemala.olx.com.gt/que-es-ubuntu-iid-146168031..

73. Martínez, Martha. Lenguaje de Programacion Ruby. [En línea] 2014. http://es.scribd.com/doc/28255974/Lenguaje-de-Programacion-Ruby.

74. Ramírez, Waldo. Net Beans Caracteríscas. [En línea] 2009. https://netbeans.org/community/releases/69/index_es.html..

75. Zapata, Javier. Niveles de pruebas de *software*. [En línea] 2013. http://pruebasdel*software*.wordpress.com/.

76. EcuRed. IDE de prgramación. [En línea] http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n#Ejemplos_de_IDE_de_programaci.C3.B3n.

77. —. Eclipse, entorno de desarrollo integrado. [En línea] http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado..

78. Pérez, J.E. Introducción a JavaScript. [En línea] http://www.librosweb.es/javascript/index.html.

79. Goñi Camejo, Ivis. *Algunas reflexiones sobre el concepto de información y sus implicaciones para el desarrollo de las ciencias de la información.* s.l. : ACIMED, 2000.

80. CABRERA, E. "Control". [En línea] 2005. www.monografias.com/trabajos14/control/control.shtml.

81. Adams, Elizabeth. *La gestión de recursos de información: importancia, desafíos y responsabilidades.* 2002.

82. *"Diseño e implementación de un sistema de información para el control del combustible en la empresa de suministros y transporte agropecuarios de Sancti Spíritus".* Comas Rodríguez, Nogueira Rivera, Gutiérrez Morales, Romero Bartutis. 144, 2011, Observatorio de la Economía Latinoamericana.

83. Vickery, Brian. *Information Science in Theory and Practice.* 1997.

84. Cohen, D. *Sistemas de Información para la toma de decisiones.* 1996.

85. Norma Cubana NC ISO/IEC.

86. HAX, A. y MAJLUF, N. *Estrategias para el Liderazgo Competitivo. De la visión a los resultados.* s.l. : Dolmen, 1997.

87. Centro del Conocimiento de Fundación EDE. *Guía para la evaluación y mejora de procesos de gestión.* 2011.

ANEXOS

Anexo 1 Transcripción de entrevista a Jorge del Sol (organizador principal de la Muestra Joven ICAIC).

1. ¿Qué es la Muestra Joven ICAIC?

Jorge del Sol: La Muestra Joven ICAIC, es un evento que reúne, anualmente, a lo más significativo de la creación audiovisual joven cubana. Un evento que pretende acercarse a los jóvenes realizadores y brindarles un espacio de exhibición, reflexión y debate. Parte de su labor es contribuir a su formación, por lo cual desarrollan talleres, conferencias y clases magistrales, tanto durante el evento como en todo el año.

2. ¿Cuáles son las secciones del evento?

Jorge del Sol: El evento cuenta con diferentes secciones como la sección de concurso, donde se presentan las obras que optan por premios en las tres categorías principales, ficción, documental y animación. Además, se otorgan premios por especialidades. La sección Haciendo Cine, reúne a proyectos que, después de participar en talleres de preparación, participan en un *pitching* con entidades que apoyan la realización audiovisual. Cada año una parte importante de las obras que consolidan apoyos en Haciendo Cine, se presentan en la Muestra Joven ICAIC, lo cual argumenta la importancia de Haciendo Cine en la producción audiovisual nacional.

También cuenta con espacios colaterales que se diseñan a partir de la concepción de cada edición: muestras homenajes, muestras internacionales, clases magistrales, conferencias, talleres, exposiciones, presentaciones de publicaciones y espacios de debates, con público y críticos, de las obras en concurso.

3. ¿Cuál es la frecuencia del evento?

Jorge del Sol: El evento se realiza anualmente, durante la primera semana de abril.

4. ¿Cuál es su alcance?

Jorge del Sol: La mayor parte de nuestro alcance está referido al público joven interesado en el audiovisual, aunque cada vez más logran acceder a otros públicos que de alguna manera intenta reflexionar en la nueva

creación audiovisual. El alcance fundamentalmente es nacional, aunque fuera de concurso se proyectan películas cuya producción pertenece a otros países.

5. ¿Qué duración tiene el festival?

Jorge del Sol: El festival dura una semana.

6. ¿Qué tiene mayor importancia dentro del evento?

Jorge del Sol: Las películas y los proyectos que se presentan al *pitching.* Quizás, el sentido más importante del evento es la exhibición de las obras de los jóvenes realizadores. Aunque el trabajo dedicado a la formación a partir de los talleres, clases magistrales y conferencias aporta texturas importantes al evento.

7. ¿Qué es *pitching*?

Jorge del Sol: *Pitching*, es la acción mediante la cual proyectos audiovisuales (o de otras especialidades artísticas) se presentan frentes a entidades o productoras capaces de asumir, total o parcialmente, la producción de una obra. *Pitching*, se le conoce en inglés, la traducción más correcta a nuestro idioma es presentación de proyectos, es una modalidad que contiene normas a seguir para presentar un proyecto de cine, fundamentalmente para obtener financiamiento, o sea dinero o servicios para garantizar en parte o completamente la producción de un cortometraje, un mediometraje, o un largometraje.

8. ¿Cómo gestionan la información?

Jorge del Sol: La información se gestiona en los espacios y medios convencionales. Cada vez más intentamos acercarnos a los nuevos medios de comunicación intentando acceder a públicos más amplios. Una parte importante de la documentación está generada y resguardada desde el Bisiesto, publicación que desde hace varios años acompaña la Muestra Joven ICAIC y donde se recoge desde la crítica joven y especializada las diferentes secciones del evento, obras y exposiciones. Otra forma de gestionar es haciendo búsquedas manuales en catálogos físicos, documentos en formato Word, Excel, pdf y en las fichas técnicas impresas y hasta haciendo uso de la memoria.

9. ¿Dónde la almacenan?

El trabajo de archivo se realiza en dos escenarios fundamentales, uno, la videoteca donde se guardan todas las obras inscritas y seleccionadas a concurso, la cual, está abierto a la consulta. Por otra parte, está la

documentación que genera el evento, la cual se almacena físicamente y también digital. Se encuentra dispersa en varios dispositivos físicos y digitales (laptops, discos duros, computadoras de la oficina), carpetas que contienen papeles, listados impresos.

10. ¿Dónde almacenan las películas?

Jorge del Sol: Las películas se almacenan en la Videoteca Joven ICAIC. Es un local destinado a ese fin el cual llamamos videoteca, las mismas se almacenan en formatos diversos: Beta, VHS, miniDV, CD, DVD, BluRay.

11. ¿Cómo están clasificadas las películas?

Jorge del Sol: Las películas se almacenan por Muestra, en las categorías: Concurso, Fuera de Concurso, Presentaciones Especiales, La mirada del Otro e Informativas, hay un poco de confusión en la forma de estructurar la clasificación.

12. ¿Actualmente emplean alguna herramienta informática para almacenar la información del evento?

Jorge del Sol: Actualmente, utilizamos las herramientas del Office para almacenar las informaciones generadas por la Muestra, lo cual representa un problema importante, ya que son un grupo de documentos aislados los cuales no son funcionales a la hora de reclamar información.

13. ¿Asignan algún presupuesto para la informatización del evento?

Jorge del Sol: Estamos solicitando un especialista en calidad de servicio social o tesis de licenciatura que pueda hacerse cargo de la informatización del evento, aun no existe.

14. ¿Qué información están interesados en almacenar en el sistema?

Jorge del Sol: Estamos interesados en almacenar, fundamentalmente, la información que genera el evento. Obras que participan en las diferentes secciones, actividades colaterales (exposiciones, personal, etc.), además de un grupo de informaciones que sirvan de referencia a los organizadores del evento y a los investigadores, periodistas y realizadores, por ejemplo la edición del evento número 13, de ella necesitaríamos tener los datos específicos del comité organizador, todas las gráficas que se crearon para dicha edición, el periódico Bisiesto correspondiente, así como el catálogo, las fechas en las que se desarrolló el evento, las películas que participaron en el concurso de jóvenes realizadores, los carteles que participaron

en el concurso de jóvenes diseñadores, las películas que están fuera de competencia en los distintos espacios de proyección creados, en la sección Haciendo Cine, almacenar todos los proyectos que participaron en el *pitching*, los datos de los jurados que integran la edición, también deseamos almacenar los datos de los eventos colaterales que tuvieron lugar en esta edición.

Anexo 2 Planificación de tareas por *Sprint*.

Tabla 10 *Sprint* 1.

Sprint 1				
Requisito	**Tarea**	**Persona**	**Estado**	**Tiempo**
RF2T1	Implementar funcionalidad Insertar Muestra	Yohan González Espinosa	Resuelta	30
RF2T2	Implementar funcionalidad Modificar Muestra	Yohan González Espinosa	Resuelta	30
RF2T3	Implementar funcionalidad Listar Muestra	Yohan González Espinosa	Resuelta	30
RF2T4	Implementar funcionalidad Eliminar Muestra	Yohan González Espinosa	Resuelta	30
RF2T5	Diseñar las interfaces necesarias para Gestionar Muestra	Yohan González Espinosa	Resuelta	40

Tabla 11 *Sprint* 2.

Sprint 2				
Requisito	**Tarea**	**Persona**	**Estado**	**Tiempo**
RF3T1	Implementar funcionalidad Insertar Película	Yohan González Espinosa	Resuelta	30
RF3T2	Implementar funcionalidad Modificar Película	Yohan González Espinosa	Resuelta	30
RF3T3	Implementar funcionalidad Listar Película	Yohan González Espinosa	Resuelta	30
RF3T4	Implementar funcionalidad Eliminar Película	Yohan González Espinosa	Resuelta	30
RF3T5	Diseñar las interfaces necesarias para Gestionar Película	Yohan González Espinosa	Resuelta	40

Tabla 12 *Sprint* 3.

Sprint 3				
Requisito	**Tarea**	**Persona**	**Estado**	**Tiempo**
RF4T1	Implementar funcionalidad Insertar Carteles	Yohan González Espinosa	Resuelta	23
RF4T2	Implementar funcionalidad Modificar Carteles	Yohan González Espinosa	Resuelta	23
RF4T3	Implementar funcionalidad Listar Carteles	Yohan González Espinosa	Resuelta	23
RF4T4	Implementar funcionalidad Eliminar Carteles	Yohan González Espinosa	Resuelta	23
RF4T5	Diseñar las interfaces necesarias para Gestionar Carteles	Yohan González Espinosa	Resuelta	31
RF5T5	Implementar funcionalidad Insertar Proyecto	Yohan González Espinosa	Resuelta	13
RF5T6	Implementar funcionalidad Modificar Proyecto	Yohan González Espinosa	Resuelta	13
RF5T7	Implementar funcionalidad Listar Proyecto	Yohan González Espinosa	Resuelta	13
RF5T8	Implementar funcionalidad Eliminar Proyecto	Yohan González Espinosa	Resuelta	13
RF5T9	Diseñar las interfaces necesarias para Gestionar Proyecto	Yohan González Espinosa	Resuelta	21

Tabla 13 *Sprint* 4.

Sprint 4				
Requisito	**Tarea**	**Persona**	**Estado**	**Tiempo**
RF6T1	Implementar funcionalidad de autenticación de usuarios	Yohan González Espinosa	Resuelta	25
RF6T2	Diseñar las interfaces requeridas para la funcionalidad autenticación de usuario.	Yohan González Espinosa	Resuelta	35
RF7T3	Implementar funcionalidad Insertar Evento Colateral	Yohan González Espinosa	Resuelta	10
RF7T4	Implementar funcionalidad Modificar Evento Colateral	Yohan González Espinosa	Resuelta	10
RF7T5	Implementar funcionalidad Listar Evento Colateral	Yohan González Espinosa	Resuelta	10
RF7T6	Implementar funcionalidad Eliminar Evento Colateral	Yohan González Espinosa	Resuelta	10
RF7T7	Diseñar las interfaces necesarias para Gestionar Evento Colateral	Yohan González Espinosa	Resuelta	20
RF8T7	Implementar funcionalidad Insertar Exposiciones	Yohan González Espinosa	Resuelta	8
RF8T8	Implementar funcionalidad Modificar Exposiciones	Yohan González Espinosa	Resuelta	8
RF8T9	Implementar funcionalidad Listar Exposiciones	Yohan González Espinosa	Resuelta	8
RF8T10	Implementar funcionalidad Eliminar Exposiciones	Yohan González Espinosa	Resuelta	8
RF8T11	Diseñar las interfaces necesarias para Gestionar Exposiciones.	Yohan González Espinosa	Resuelta	16

Tabla 14 *Sprint* 5.

Sprint 5				
Requisito	**Tarea**	**Persona**	**Estado**	**Tiempo**
RF9T1	Implementar funcionalidad Cargar Archivos.	Yohan González Espinosa	Resuelta	1
RF9T2	Diseñar las interfaces requeridas para la funcionalidad Cargar Archivos.	Yohan González Espinosa	Resuelta	1
RF10T1	Implementar funcionalidad Exportar Listado.	Yohan González Espinosa	Resuelta	1
RF10T2	Diseñar las interfaces requeridas para la funcionalidad Exportar Listado.	Yohan González Espinosa	Resuelta	1
RF11T1	Implementar funcionalidad Buscar Contenido	Yohan González Espinosa	Resuelta	1
RF11T2	Diseñar las interfaces requeridas para la funcionalidad Buscar Contenido.	Yohan González Espinosa	Resuelta	1

Anexo 3 Descripción del Modelo de Datos.

Tabla 15 Descripción tabla pelicula.

Nombre Tabla: pelicula		
Descripción: En esta tabla se almacenan los datos correspondientes a una película		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador de la película Es la llave primaria de la tabla.
codigo	Int	Codificación creada para almacenar las películas
titulo	varchar	Es el título de la película almacenada.
fecha	date	Fecha de realización de la película.
nacionalidad	varchar	Nacionalidad de la película.
tiempo	varchar	Tiempo de duración de la película.
fotograma	varchar	Es una imagen de la película.
cartel	varchar	Cartel de la película
muestraid	Int	Llave foránea de la relación existente entre la tabla muestra y la tabla pelicula
muestra_concursomuestraid	Int	Llave foránea de la relación existente entre la tabla muestraconcurso y la tabla pelicula.

Tabla 16 Descripción tabla muestraconcurso.

Nombre Tabla: muestraconcurso

Descripción: Esta tabla es la que se genera al existir relación de muchos a muchos entre muestra y concurso.		
Atributos	**Tipo**	**Descripción**
muestraid	Int	Llave foránea de la relación existente entre la tabla muestraconcurso y la tabla muestra.
concursoid	Int	Llave foránea de la relación existente entre la tabla muestraconcurso y la tabla concurso.

Tabla 17 Descripción tabla concurso.

Nombre Tabla: concurso		
Descripción: En esta tabla se almacenan los datos correspondientes a un concurso.		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador del concurso. Es la llave primaria de la tabla.
desc	varchar	Descripción del concurso.

Tabla 18 Descripción tabla muestrapersona.

Nombre Tabla: muestrapersona		
Descripción: Esta tabla es la que se genera al existir relación de muchos a muchos entre muestra y persona.		
Atributos	**Tipo**	**Descripción**
muestraid	Int	Llave foránea de la relación existente entre la tabla muestrapersona y la tabla muestra.
personaid	Int	Llave foránea de la relación existente entre la tabla muestrapersona y la tabla persona.
rol	Int	Rol en el que se desempeña el artista.

Tabla 19 Descripción tabla persona.

Nombre Tabla: persona		
Descripción: En esta tabla se almacenan los datos correspondientes a una persona		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador de la persona. Es la llave primaria de la tabla.
ci	Int	Número de carnet de identidad de la persona.
nombre	varchar	Nombre de la persona.
apellido	varchar	Apellido de la persona.
sexo	char	´f´ para femenino, ´m´ para masculino.
ciudad	varchar	Ciudad a la cual pertenece la persona.

formacion	varchar	Formación que tenga la persona, escuela de la cual haya egresado.
especialidad	Int	Especialidad de cine a la que se dedica.

Tabla 20 Descripción tabla exposicion.

Nombre Tabla: exposicion		
Descripción: En esta tabla se almacenan los datos correspondientes a una exposición.		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador de la exposición Es la llave primaria de la tabla.
nombre	varchar	Nombre de la exposición.
lugar	varchar	Lugar donde se realiza la exposición.
tipo	varchar	Tipo de exposición.
muestraid	Int	Llave foránea proveniente de la tabla muestra

Tabla 21 Descripción tabla exposicion_persona.

Nombre Tabla: exposicion_persona		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla persona y la tabla exposicion.		
Atributos	**Tipo**	**Descripción**
exposicionid	Int	Llave foránea proveniente de la tabla exposicion.
personaid	Int	Llave foránea proveniente de la tabla persona.
rol	Int	Rol en el que se desempeña el artista.

Tabla 22 Descripción tabla muestra_jurado.

Nombre Tabla: muestra_jurado		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla muestra y la tabla jurado.		
Atributos	**Tipo**	**Descripción**
muestraid	Int	Llave foránea proveniente de la tabla muestra.
juradopersonaid	Int	Llave foránea proveniente de la tabla jurado.

Tabla 23 Descripción tabla jurado.

Nombre Tabla: jurado		
Descripción: En esta tabla se almacenan los datos correspondientes a un jurado.		
Atributos	**Tipo**	**Descripción**

personaid	Int	Llave foránea proveniente de la tabla persona.
tipo	Int	Especialidad de cine que evalúa.

Tabla 24 Descripción tabla pelicula_persona.

Nombre Tabla: pelicula_persona		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla persona y la tabla pelicula.		
Atributos	**Tipo**	**Descripción**
peliculaid	Int	Llave foránea proveniente de la tabla película.
personaid	Int	Llave foránea proveniente de la tabla persona.
rol	Int	Rol que desempeña la persona en la película.

Tabla 25 Descripción tabla evento_colateral.

Nombre Tabla: evento_colateral		
Descripción: En esta tabla se almacenan los datos correspondientes a un evento colateral		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador del evento colateral. Es la llave primaria de la tabla.
tipo	varchar	Tipo de evento colateral.
des	varchar	Descripción del evento.
muestraid	varchar	Llave foránea proveniente de la tabla muestra.

Tabla 26 Descripción tabla muestra_premio.

Nombre Tabla: muestra_premio		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla muestra y la tabla premio		
Atributos	**Tipo**	**Descripción**
muestraid	Int	Llave foránea proveniente de la tabla muestra.
premioid	Int	Llave foránea proveniente de la tabla premio.

Tabla 27 Descripción tabla premio.

Nombre Tabla: premio		
Descripción: En esta tabla se almacenan los datos correspondientes a un premio		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador del premio. Es la llave primaria de la tabla.

desc	varchar	Descripción del premio.

Tabla 28 Descripción tabla muestra_entidad.

Nombre Tabla: muestra_entidad		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla muestra y la tabla entidad		
Atributos	**Tipo**	**Descripción**
muestraid	Int	Llave foránea proveniente de la tabla muestra.
entidadid	Int	Llave foránea proveniente de la tabla entidad.

Tabla 29 Descripción tabla entidad.

Nombre Tabla: entidad		
Descripción: En esta tabla se almacenan los datos correspondientes a una entidad		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador de la entidad. Es la llave primaria de la tabla.
desc	varchar	Descripción de la entidad.

Tabla 30 Descripción tabla bisiesto.

Nombre Tabla: bisiesto		
Descripción: En esta tabla se almacenan los datos correspondientes a un Bisiesto.		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador del Bisiesto. Es la llave primaria de la tabla.
file	varchar	Es el periódico Bisiesto en formato pdf.
muestraid	Int	Llave foránea proveniente de la tabla muestra.

Tabla 31 Descripción tabla persona_bisiesto.

Nombre Tabla: persona_bisiesto		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla persona y la tabla bisiesto		
Atributos	**Tipo**	**Descripción**
personaid	Int	Llave foránea proveniente de la tabla persona.
bisiestoid	Int	Llave foránea proveniente de la tabla bisiesto.

Tabla 32 Descripción tabla jurado.

Nombre Tabla: usuario		
Descripción: En esta tabla se almacenan los datos correspondientes a un usuario.		
Atributos	**Tipo**	**Descripción**
personaid	Int	Llave foránea proveniente de la tabla persona.
nombre_usuario	varchar	Nombre de usuario de la persona.
contrasena	varchar	Contraseña del usuario.

Tabla 33 Descripción tabla guion.

Nombre Tabla: guion		
Descripción: En esta tabla se almacenan los datos correspondientes a un guión.		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador de un guion. Es la llave primaria de la tabla.
muestra_concursomuestraid	varchar	Llave foránea proveniente de la tabla muestra_concurso.
muestra_concursoconcursoid	varchar	Llave foránea proveniente de la tabla muestra_concurso.
pitching	binary	

Tabla 34 Descripción tabla persona_guion.

Nombre Tabla: persona_guion		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla persona y la tabla guion		
Atributos	**Tipo**	**Descripción**
personaid	Int	Llave foránea proveniente de la tabla persona.
guionid	Int	Llave foránea proveniente de la tabla guion.

Tabla 35 Descripción tabla cartel.

Nombre Tabla: cartel		
Descripción: En esta tabla se almacenan los datos correspondientes a un cartel		
Atributos	**Tipo**	**Descripción**
id	Int	Es el identificador de un cartel. Es la llave primaria de la tabla.
muestra_concursomuestraid	varchar	Llave foránea proveniente de la tabla muestra_concurso.
muestra_concursoconcursoid	varchar	Llave foránea proveniente de la tabla muestra_concurso.
nombre	varchar	Nombre del cartel.

Tabla 36 Descripción tabla premio_cartel.

Nombre Tabla: premio_cartel		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla premio y la tabla cartel		
Atributos	**Tipo**	**Descripción**
premioid	Int	Llave foránea proveniente de la tabla premio.
cartelid	Int	Llave foránea proveniente de la tabla cartel.

Tabla 37 Descripción tabla premio_pelicula.

Nombre Tabla: premio_pelicula		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla premio y la tabla pelicula		
Atributos	**Tipo**	**Descripción**
premioid	Int	Llave foránea proveniente de la tabla premio.
peliculaid	Int	Llave foránea proveniente de la tabla pelicula.

Tabla 38 Descripción tabla cat_tematica.

Nombre Tabla: cat_tematica		
Descripción: En esta tabla se almacena los datos referentes a las temáticas.		
Atributos	**Tipo**	**Descripción**
id	Int	Llave primaria de la tabla tematica.
desc	Int	Descripción de una temática.

Tabla 39 Descripción tabla productora.

Nombre Tabla: productora		
Descripción: En esta tabla se almacena los datos referentes a las productoras.		
Atributos	**Tipo**	**Descripción**
id	Int	Llave primaria de la tabla productora.
desc	Int	Descripción de la productora.

Tabla 40 Descripción tabla formato.

Nombre Tabla: formato		
Descripción: Esta tabla se almacena los datos referentes a los formatos de las películas.		
Atributos	**Tipo**	**Descripción**
id	Int	Llave primaria de la tabla formato.
desc	Int	Descripción de un formato.

Tabla 41 Descripción tabla pelicula_cat_tematica.

Nombre Tabla: película_cat_tematica		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla pelicula y la tabla cat_tematica.		
Atributos	**Tipo**	**Descripción**
peliculaid	Int	Llave foránea proveniente de la tabla pelicula
cat_tematicaid	Int	Llave foránea proveniente de la tabla cat_tematica.

Tabla 42 Descripción tabla pelicula_productora.

Nombre Tabla: película_productora		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla pelicula y la tabla productora.		
Atributos	**Tipo**	**Descripción**
peliculaid	Int	Llave foránea proveniente de la tabla pelicula.
productoraid	Int	Llave foránea proveniente de la tabla productora.

Tabla 43 Descripción tabla pelicula_formato.

Nombre Tabla: película_formato		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla pelicula y la tabla formato.		
Atributos	**Tipo**	**Descripción**
peliculaid	Int	Llave foránea proveniente de la tabla pelicula.
formatoid	Int	Llave foránea proveniente de la tabla formato.

Tabla 44 Descripción tabla persona_cartel.

Nombre Tabla: persona_cartel		
Descripción: Esta tabla es la resultante de la relación de muchos a muchos entre la tabla persona y la tabla cartel.		
Atributos	**Tipo**	**Descripción**
personaid	Int	Llave foránea proveniente de la tabla persona.
cartelid	Int	Llave foránea proveniente de la tabla cartel.

Tabla 45 Descripción tabla categoria.

Nombre Tabla: categoria
Descripción: Esta tabla se almacena los datos referentes a las categorías de las películas.

Atributos	Tipo	Descripción
id	Int	Llave primaria de la tabla categoria.
desc	Int	Descripción de una categoría.
peliculaid	Int	Llave foránea proveniente de la tabla pelicula.

Anexo 4 Especificación de requisitos de *software*

Se muestra a continuación el artefacto Especificación de requisitos de *software* firmado por el cliente.

UCI
Universidad de las Ciencias Informáticas

Especificación de requisitos de software.

Sistema de Gestión de Información para la Muestra Joven ICAIC.

Universidad de las Ciencias Informáticas
Carretera a San Antonio Km 2 ½ , Torrens. Boyeros, Ciudad de La Habana. Cuba
Teléfono: + 53 (7) 837 2402 Fax: + 53 (7) 837 2407 E-mail: ip@uci.cu

Control del Documento

Título: Especificación de requisitos de software.
Versión: 1.0

	Nombre	Cargo
Elaborado por:	Yohan González Espinosa	Líder del proyecto
Aprobado por:	Jorge del Sol	Firma:
Cargo:	Coordinador general	Fecha: 16/11/2014

Reglas de Confidencialidad

Clasificación: USO INTERNO.
Forma de distribución: PDF.

Control de Cambios

Versión	Sección, Figura, Tabla	Tipo*	Fecha	Autor del cambio	Descripción del cambio
0.1	Todo el documento	A	9/11/2014	Yohan González Espinosa	Elaboración del documento.

* Indicar el tipo de cambio: **A** Alta; **B** Baja; **M** Modificación

1 Introducción

1.1 Objetivo

Describir el entendimiento común alcanzado por los involucrados respecto a los requisitos del Sistema de Gestión de Información para la Muestra Joven ICAIC.

1.2 Alcance

Este documento contiene los requisitos funcionales y no funcionales correspondientes al Sistema de Gestión de Información para la Muestra Joven ICAIC.

1.3 Definiciones y acrónimos

ICAIC: Instituto Cubano de Arte e Industria Cinematográficos.

CASE: Ingeniería de Software Asistida por Computadora, por sus siglas en inglés *Computer Aided Software Engineering*

SO: Sistema Operativo

RAM: Memoria de Acceso Aleatorio, en inglés: *Random-Access Memory*.

MB: *Megabyte*

GB: *Gigabyte*

2 Vista del Sistema de Gestión de Información para la Muestra Joven ICAIC

2.1 Descripción del sistema

El Sistema de Gestión de Información permitirá automatizar el proceso de gestión de información en la oficina de la Muestra Joven ICAIC, facilitando, el análisis de los datos de la muestra, la evolución de sus participantes, la sistematización de sus *records*; y permitirá el registro de una codificación que identifique las películas de la videoteca.

3 Catálogo de requisitos

3.1.3 Requisitos funcionales

Nº	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF1	Gestionar Persona.	Insertar, modificar, listar, eliminar datos de todos los participantes de la Muestra Joven ICAIC.	Alta	Media
RF2	Gestionar Muestra Joven.	Insertar, modificar, listar, eliminar datos relacionados con la Muestra Joven ICAIC. Los datos que se manejan son: ✓ Número de la muestra. ✓ Fecha en que se realizó la muestra. ✓ Trípticos, posters, periódicos y catálogos en formato pdf. ✓ Jurados. ✓ Patrocinadores.	Alta	Media

RF3	Gestionar Película.	Insertar, modificar, listar, eliminar datos relacionados con las películas de las distintas secciones. Los datos asociados a las películas son: ✓ Código que representa su ubicación en la videoteca. ✓ Título. ✓ Año de exhibición. ✓ Nacionalidad. ✓ Reparto.	Alta	Media
RF4	Gestionar Carteles.	Insertar, modificar, listar, eliminar datos relacionados con los carteles en concurso. Los datos relacionados a los carteles son: ✓ Nombre de la película. ✓ Diseñador.	Alta	Media

RF5	Gestionar Proyecto.	Insertar, modificar, listar, eliminar datos relacionados con los proyectos participantes en la sección Haciendo Cine. Los datos almacenados son: ✓ Nombre. ✓ Director. ✓ Productor. ✓ Guionista.	Alta	Media
RF6	Autenticar Usuario.	El usuario debe autenticarse para comprobar permisos de acceso.	Media	Media
RF7	Gestionar Eventos Colaterales.	Insertar, modificar, listar, eliminar datos relacionados con los eventos colaterales (talleres, conferencias, homenajes). Los datos gestionados son: ✓ Tipo de evento. ✓ Descripción.	Media	Media

		✓ Personas.		
RF8	Gestionar Exposiciones.	Se cargan los archivos (.jpeg, .png, .pdf) relacionados a la muestra y al concurso de carteles.	Media	Media
RF9	Cargar archivos.	Se exportan los listados en formato .xls.	Baja	Media
RF10	Exportar listados.	✓ Buscar contenido en todas las tablas de la base de datos mostrando la incidencia de la búsqueda en cada una de ellas. ✓ Buscar resultados en una tabla específica mediante filtros.	Baja	Media
RF11	Buscar contenido.	Se cargan los archivos (.jpeg, .png, .pdf) relacionados a la muestra y al concurso de carteles.	Baja	Media

3.2 Requisitos no funcionales

3.2.1 Descripción de requisitos no funcionales

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	Comprensibilidad Instructibilidad Operabilidad
Objetivo	El tiempo de entrenamiento requerido para que usuarios normales y avanzados sean productivos operando el sistema es de 2 días.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Usabilidad
Sub-atributos/Sub-características	Operabilidad Atractivo Conformidad
Objetivo	Debe poseer una interfaz agradable para el cliente.
Origen	El sistema
Artefacto	Interfaz de usuario

Atributo de Calidad	Fiabilidad
Sub-atributos/Sub-características	Fiabilidad
Objetivo	El sistema estará disponible 24 horas del día, los siete días de la semana.
Origen	El sistema
Artefacto	El sistema

Atributo de Calidad	Fiabilidad
Sub-atributos/Sub-características	Tolerancia al defecto Recuperatibilidad
Objetivo	La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores.
Origen	Implementación
Artefacto	El sistema

Atributo de Calidad	Eficiencia
Sub-atributos/Sub-características	Comportamiento en el tiempo

Objetivo	Tiempo de respuesta promedio de las peticiones que se realizan al servidor no deberá ser mayor de 30 segundos.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Eficiencia
Sub-atributos/Sub-características	Eficacia
Objetivo	El número de clientes o transacciones que el sistema puede alojar es de 2000.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Seguridad
Sub-atributos/Sub-características	Estabilidad
Objetivo	Protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Seguridad
Sub-atributos/Sub-características	Estabilidad
Objetivo	El sistema debe mantener en todo momento la seguridad de la información asegurando la autenticidad de la misma.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Seguridad
Sub-atributos/Sub-características	Disponibilidad
Objetivo	El sistema debe garantizar la confidencialidad, integridad y disponibilidad de la información que se procese en el sistema.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Seguridad
Sub-atributos/Sub-características	Estabilidad
Objetivo	El control de acceso se establecerá por roles que se le asignarán a los usuarios que interactúen con el sistema.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Soporte
Sub-atributos/Sub-características	Mantenibilidad
Objetivo	Soporte para grandes volúmenes de datos y velocidad de procesamiento.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Soporte
Sub-atributos/Sub-características	Mantenibilidad Estabilidad
Objetivo	Tiempo de respuesta rápido en accesos concurrentes.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Restricciones de diseño
Sub-atributos/Sub-características	Precisión
Objetivo	El lenguaje de programación es PHP 5.3.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Restricciones de diseño
Sub-atributos/Sub-características	Precisión
Objetivo	El marco de trabajo de desarrollo es Symfony 2.3, Bootstrap 3.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Restricciones de diseño
Sub-atributos/Sub-características	Precisión
Objetivo	La herramienta IDE de desarrollo utilizada será phpStorm 8.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Restricciones de diseño
Sub-atributos/Sub-características	Precisión
Objetivo	La herramienta *CASE* utilizada es Visual Paradigm para UML 8.0 para modelado.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Restricciones de diseño
Sub-atributos/Sub-características	Precisión
Objetivo	La herramienta gestora de base de datos es PostgreSQL 9.1.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Restricciones de diseño
Sub-atributos/Sub-características	Precisión
Objetivo	Servidor web Apache 2.4.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Interfaz
Sub-atributos/Sub-características	Comprensibiliad
Objetivo	El sistema tiene que ofrecer una interfaz amigable, fácil de operar.
Origen	Ser humano
Artefacto	Interfaz de usuario

Atributo de Calidad	Interfaz
Sub-atributos/Sub-características	Comprensibiliad
Objetivo	Los colores a utilizar deben ser rojo, gris, blanco y negro.
Origen	Ser humano
Artefacto	Interfaz de usuario

Atributo de Calidad	Interfaz
Sub-atributos/Sub-características	Comprensibiliad
Objetivo	El software debe tener el nombre del evento ("Muestra Joven ICAIC") y su logo.
Origen	Ser humano
Artefacto	Interfaz de usuario

Atributo de Calidad	Software
Sub-atributos/Sub-características	Idoneidad
Objetivo	Sistema Operativo (SO) en el servidor: GNU/Linux Ubuntu 12.04 o similares, por recomendación.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Software
Sub-atributos/Sub-características	Idoneidad
Objetivo	SO en las máquinas clientes: cualquiera que decida el cliente.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Hardware
Sub-atributos/Sub-características	Idoneidad
Objetivo	Se recomienda para el funcionamiento mínimo máquinas clientes con procesador Pentium II o versiones superiores con no menos de 256 MB de RAM.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Hardware
Sub-atributos/Sub-características	Idoneidad
Objetivo	Se recomienda para el funcionamiento óptimo máquinas clientes con procesadores Pentium IV y 512 MB de RAM.
Origen	Ser humano
Artefacto	El sistema

Atributo de Calidad	Hardware
Sub-atributos/Sub-características	Idoneidad
Objetivo	El procesador para el funcionamiento óptimo del sistema en el servidor no debe ser menor a 2 GB de RAM.
Origen	Ser humano
Artefacto	El sistema

4 Anexos

4.1 Participantes en el proyecto

Nombres y Apellidos	Rol	Área funcional	Artefacto involucrado	Observaciones
Yohan González	Analista	Receptor	Especificación de requisitos.	Autor de los artefactos

Espinosa			Modelo de dominio Carta de aceptación.	
Jorge del Sol	Cliente	Proveedor	Especificación de requisitos. Modelo de dominio Carta de aceptación.	Aprobó el documento. Aportó la información

Anexo 5 Acta de aceptación.

UCI Universidad de las Ciencias Informáticas

Acta de aceptación

Sistema de Gestión de Información para la Muestra Joven ICAIC

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

Autor: Yohan González Espinosa

ACTA DE ACEPTACIÓN

En cumplimiento de la ejecución del proyecto: Sistema de gestión de información para la Muestra Joven ICAIC, se hace entrega de los productos que se relacionan a continuación:

- Especificación de Requisitos de Software.
- Sistema de gestión de información para la Muestra Joven ICAIC.
- Modelo de dominio.
- Modelo de despliegue.

La Parte Cliente, luego de haber revisado los productos de trabajo determina que **acepta** la solución informática desarrollada para la Muestra Joven ICAIC, pues satisface en su totalidad los requerimientos iniciales pactados.

Como constancia de la validez de la presente firman en dos (2) ejemplares, al mismo tenor, en La Habana, a los 16 días del mes de marzo de 2015.

Entrega: Universidad de las Ciencias Informáticas	**Recibe:** Instituto Cubano de Arte e Industria Cinematográfico
Nombre y apellidos: Yohan González Espinosa	**Nombre y apellidos:** Marisol Rodríguez Rosabal
Cargo: Estudiante	**Cargo:** Directora Muestra Joven ICAIC
Firma:	**Firma:**