

PLANIFICACIÓN DEL PROYECTO

Introducción

Alcance

El sistema es un software educativo que permite modificar y simular ejercicios de la Cátedra de Física General I, de manera específica en Física/Mecánica Newtoniana, en el área de Cinemática (en una y dos dimensiones), que permita la comprensión del contenido por parte del alumno.

Este sistema permitirá realizar de forma automática y desde cualquier lugar conectado a Internet; los siguientes procesos:

Gestión de usuarios: permite el registro de nuevos usuarios al sistema así como también la modificación posterior de sus datos y su posible eliminación.

Gestión de proyectos: los proyectos son los ejercicios que los profesores podrán publicar en el sistema para que los alumnos accedan a ellos. Podrán ser insertados, modificados y/o eliminados.

Gestión de teoría: todo el material teórico que el sistema pueda ofrecer será administrado por los profesores que podrán adicionar nuevo contenido, modificar el existente o eliminarlo.

El alcance del proyecto se verá afectado por lo que aquí se incluya.

Definiciones, Acrónimos y Abreviaturas

En esta sección se presentan definiciones que permiten comprender de una forma clara la terminología que se encuentra en cada elemento del documento.

Arquitectura de Software:

Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información.

La Arquitectura de Software establece los fundamentos para que analistas, diseñadores, programadores, etc. trabajen en una línea común que permita alcanzar los objetivos del sistema de información, cubriendo todas las necesidades.

Ingeniería de Requerimientos:

Comprende todas las tareas relacionadas con la determinación de las necesidades o de las condiciones a satisfacer para un software nuevo o modificado, tomando en cuenta los diversos requerimientos de los inversores, que pueden entrar en conflicto entre ellos.

Requerimiento de Sistemas:

Es una necesidad documentada sobre el contenido, forma o funcionalidad de un producto o servicio. Se usa en un sentido formal en la ingeniería de sistemas o la ingeniería de software.

Artefactos:

El término Artefacto, en conexión con el desarrollo de software, está mayormente asociado a métodos o procesos de desarrollo específicos, como el Proceso Unificado.

Un artefacto es un producto tangible resultante del proceso de desarrollo de software. Algunos artefactos como los casos de uso, diagrama de clases u otros modelos UML ayudan a la descripción de la función, la arquitectura o el diseño del software.

Caso de Uso:

Es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización de software. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico.

UML:

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

Requerimientos Funcionales:

Define el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica. Son complementados por los requerimientos no funcionales, que se enfocan en cambio en el diseño o la implementación.

Requerimientos No Funcionales:

Es un requerimiento que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requerimientos funcionales.

Los requerimientos no funcionales más habituales son la estabilidad, la portabilidad y el costo.

Entorno y fundamentos

Identificación del Problema

Actualmente no existe una aplicación de software que permita a los estudiantes de Física General I de la escuela de Ingeniería Informática de la Universidad Católica Andrés Bello, reforzar los conocimientos adquiridos en clase.

El estudio de la materia se hace, principalmente por medio de guías de estudio y libros de texto, que en muchas ocasiones, su manejo, comprensión y entendimiento puede resultar complejo.

Es importante que las aplicaciones de software se hagan cada vez más presentes en los primeros semestres de la carrera para introducir paulatinamente al estudiante en el uso de la tecnología en sus técnicas de estudio.

El Proyecto

Objetivo

Desarrollar un software educativo para apoyar el aprendizaje de la Cátedra de Física General I en estudiantes del ciclo básico de la escuela de Ingeniería Informática.

Alcance

EN EL ALCANCE

- *Crear una aplicación Web para usar con servidores estándar para Web y aplicaciones*
- *Que funcione en los navegadores más populares (IE6 o posterior, Mozilla Firefox 3 o posterior, Google Chrome)*
- *Seguridad para cuentas de usuario, contraseñas y permisos*

- *Que la aplicación ofrezca un módulo de exámenes*

FUERA DEL ALCANCE

- *Crear un nuevo servidor Web o de aplicaciones*
- *Que funcione en navegadores poco comunes o antiguos*
- *Seguridad especial contra hackers. Instalar o parchar agujeros de seguridad en los componentes de software utilizados.*
- *Que los docentes puedan agregar nuevos exámenes.*

Justificación

La creciente utilización de las tecnologías de información en casi todos los aspectos de la vida diaria y la necesidad de hacer el proceso de enseñanza más ágil e interactivo, fundamentan las bases del desarrollo de software educativo (SE). Un SE es un componente cuyas características estructurales y funcionales sirven de apoyo al proceso de enseñar y aprender.

Para el desarrollo de un SE, es necesario considerar la motivación e interés, evitando que el proceso sea largo y complicado tanto para los alumnos como para los profesores. En la actualidad, la idea de utilizar recursos adicionales a los dados en clase que permitan reforzar y colaborar con el aprendizaje, son positivamente recibidos por todas las partes. Con esto es factible pensar que si se presenta al alumnado un recurso que les permita comprender, asimilar y aprender el contenido de la cátedra de Física General I, así como a los profesores un recurso que los ayude en el proceso de enseñanza, se estimulará el uso de este SE y se extenderán sus aplicaciones y campos de acción.

Características y Beneficios

Una aplicación para Web reutilizable con funcionalidad para crear, editar, borrar y consultar usuarios, material teórico y especialmente ejercicios prácticos. Esto automatiza todas las operaciones de los usuarios y asegura que podrán encontrar siempre información que de forma automática se encuentre actualizada.

La aplicación del sitio deberá tener una apariencia altamente configurable que le permita ajustarse a la apariencia y carácter del cliente. Esto permite reutilizar la aplicación del sitio para que tenga una apariencia de alta calidad que pueda ser muy buena.

La aplicación del sitio deberá ser segura y sólo permitir usuarios con los permisos adecuados para editar, borrar o añadir otros usuarios al sistema. Esto para prevenir trampas o la publicación de información falsa.

Resumen de la Metodología

Herramientas de Desarrollo y Colaboración

HERRAMIENTA	FUENTE	CANTIDAD	ESTADO	COMENTARIOS
<i>Materiales de Entrenamiento</i>	<i>Libro/Curso en una tecnología específica</i>	<i>1</i>	<i>Localizado</i>	<i>Consultas en línea.</i>
<i>Estaciones de Trabajo para Desarrollo</i>	<i>PC 512 MB RAM Exploradores con Flash Player 9 ó posterior.</i>	<i>3</i>	<i>Cumplen</i>	<i>El grupo de desarrollo utilizará equipo existente</i>
<i>Licencias de IDE</i>	<i>Licencias de desarrollo estándar</i>	<i>N/A</i>	<i>Cumple</i>	<i>Utilizaremos herramientas open source</i>

<i>Licencias de Herramientas para Pruebas</i>	<i>Licencias de desarrollo estándar</i>	<i>N/A</i>	<i>Cumple</i>	<i>Utilizaremos herramientas open source</i>
---	---	------------	---------------	--

Tabla D1. Herramientas de Desarrollo y Colaboración. Fuente: Elaboración propia.

El software estará desarrollado en FLEX bajo tecnología JAVA con base de datos MySQL.

Fases e iteraciones

El proyecto consta de 4 fases principales: INICIO; ELABORACIÓN, CONSTRUCCIÓN Y TRANSICIÓN. A su vez, estas fases se componen de iteraciones las cuales son previamente definidas por el equipo de desarrollo. En la tabla presentada a continuación se muestran las fases de INICIO y ELABORACIÓN con sus respectivas iteraciones.

Estas iteraciones han sido escogidas de acuerdo al nivel de riesgo ó dificultad de las actividades para llevarlas a cabo.

FASES	CANTIDAD DE ITERACIONES	COMENTARIOS
<i>INICIO</i>	<i>1 iteración</i>	<i>Es la iteración inicial, del análisis, diseño y pruebas que se obtiene de ésta fase se conceptualiza el sistema, definiendo las reglas de negocio y estableciendo el punto de partida de la segunda fase.</i>
<i>ELABORACIÓN</i>	<i>1 iteración</i>	<i>1era Iteración: se llevará a cabo el análisis, diseño, implementación y pruebas de los casos de uso: Ingresar, Modificar, Eliminar y Consultar Proyecto.</i>

Tabla D2. Fases e iteraciones. Fuente: Elaboración propia.

Estructura del Trabajo

PASO	DESCRIPCIÓN
1	Análisis
1.1	Levantamiento de información
1.1.A	Realización de Entrevistas
1.1.B	Elaboración de ER's
1.1.C	Validación de Requerimientos
1.2	Selección de la tecnología a utilizar
1.2.A	Determinación de las alternativas tecnológicas aplicables
1.2.B	Estudio de las alternativas tecnológicas
1.3	Definición estructural del sistema
1.3.A	Definición de la arquitectura del software
1.3.B	Definición de las herramientas de hardware
2	Diseño
2.1	Definición del prototipo
2.1.A	Elaboración del prototipo
2.1.B	Evaluación del prototipo
2.1.C	Elaboración del prototipo funcional
3	Implementación
3.1	Desarrollo del Módulo <i>Proyecto</i>
3.2	Desarrollo del Módulo <i>Estudiante</i>

3.3	Desarrollo del Módulo <i>Profesor</i>
3.4	Desarrollo del Módulo <i>Teoría</i>
3.5	Desarrollo del Módulo <i>Usuario</i>
3.6	Desarrollo del Módulo <i>Sección</i>
4	Pruebas
4.1	Pruebas de los módulos
4.2	Validación del sistema

Tabla D3. Estructura del trabajo. Fuente: Elaboración propia.

Entregables del Proyecto

DESCRIPCIÓN	ENTREGABLE	FECHA DE LA ENTREGA
<i>Documentos de Conceptualización del sistema. FASE DE INICIO.</i>	<i>Documento de Visión</i>	<i>Semana 2</i>
	Plan de Proyecto	<i>Semana 3</i>
	Modelo de Casos de Uso	<i>Semana 3</i>
	Estándares para el prototipo	<i>Semana 6</i>
<i>Documentos de la arquitectura del software. FASE DE ELABORACIÓN</i>	<i>Arquitectura del Software</i>	<i>Semana 11</i>
<i>FASE DE CONSTRUCCIÓN</i>	<i>Capacidad inicial del sistema.</i>	<i>Semana 28</i>
<i>FASE DE</i>	<i>Sistema funcional</i>	<i>Semana 27</i>

DESCRIPCIÓN	ENTREGABLE	FECHA DE LA ENTREGA
<i>TRANSICIÓN</i>	<i>Manuales de usuario y documentación del sistema</i>	<i>Semana 28</i>

Tabla D4. Entregables del proyecto. Fuente: Elaboración propia.

Calendario del proyecto

Esta sección muestra un calendario de las principales actividades del proyecto, incluyendo sólo fases de Inicio y Elaboración. En las tablas siguientes la fecha de aprobación indica cuándo el objeto en cuestión tiene un estado de completitud suficiente para someterse a revisión y aprobación, pero esto no quita la posibilidad de su posterior refinamiento y cambios.

DISCIPLINAS / ARTEFACTOS GENERADOS DURANTE LA FASE DE INICIO (Iteración Inicial)	COMIENZO	APROBACIÓN
Análisis		
Documento de Visión del Sistema	Semana 1	Semana 2
Plan de Proyecto	Semana 2	En cada iteración
Modelo de Casos de uso (v.1)	Semana 3	Semana 4
Glosario inicial del proyecto	Semana 3	
Actividades agregadas a RUP por la metodología utilizada.	Semana 3	Semana 8

Estudio inicial de riesgos y su evaluación	Semana 4	Semana 5
Lista de requerimientos y restricciones principales	Semana 4	Semana 5
Criterios de Evaluación basados en MOSCA	Semana 5	Semana 6
Diseño		
Modelo de Datos	Semana 5	En cada iteración
Estándares para el prototipo	Semana 6	En cada iteración
Mapa de navegación	Semana 6	Semana 7
Implementación		
Prototipo de Interfaces de Usuario	Semana 7	En cada iteración

Tabla D5. Plan de Proyecto. Fase de Inicio. Fuente: Elaboración propia.

DISCIPLINAS / ARTEFACTOS GENERADOS DURANTE LA FASE DE ELABORACIÓN (1 ITERACIÓN)	COMIENZO	APROBACIÓN
Análisis		
Especificación de los Casos de Uso	Semana 9	Siguiente fase
Actualización de plan de iteraciones	Semana 10	Siguiente fase
Lista revisada de riesgos	Semana 10	Siguiente fase
Plan de Proyecto (v.2)	Semana 10	Siguiente fase

Diseño		
Arquitectura de Software	Semana 11	En cada iteración
Refinamiento de los requerimientos de diseño gráfico de las interfaces.	Semana 12	En cada iteración
Implementación		
Inicio de desarrollo de módulos a partir de casos de uso seleccionados	Semana 12	Siguiente Fase
Pruebas		
Aplicación de pruebas unitarias	Semana 16	En cada iteración
Evaluación		
Creación y aplicación de los cuestionarios de evaluación bajo el modelo MOSCA adaptado a software educativo.	Semana 16	En cada iteración

Tabla D6. Plan de Proyecto. Fase de Elaboración. Fuente: Elaboración propia.

Manejo de Riesgos

RIESGO	CONSECUENCIAS	MITIGACIÓN DEL RIESGO
1. La poca receptividad por parte de los usuarios del software.	El proyecto no	-
2. La pérdida de interés de los usuarios a medida que utilizan el software.		-

<p>3. La no aportación de material valioso que contribuya con el proceso de enseñanza-aprendizaje.</p>	<p>cumple con sus objetivos.</p>	<p>-</p>
<p>4. Los alumnos podrían sustituir al profesor en el aula de clase con el software, lo que conlleva a disminuir el porcentaje de asistencia a clase.</p>	<p>cumple con sus objetivos.</p>	<p>Realizar reuniones con los alumnos para señalar los objetivos del sistema y disuadir de la posibilidad de que no es necesaria la asistencia a clases, ya que son éstas quienes proporcionan las bases mínimas para el uso del sistema.</p>
<p>5. La poca disponibilidad para enriquecer y mantener al día el contenido del software por parte de los profesores.</p>	<p>Abandono de la herramienta, provocando su expiración.</p>	<p>Realizar reuniones con los involucrados para señalar la importancia de mantener al día el sistema.</p>
<p>6. El sistema no resulta fácil de usar ni de aprender.</p>	<p>Abandono del uso del sistema.</p>	<p>Realizar demostraciones con los usuarios, inducciones y/o proporcionar manuales de usuario.</p>
<p>7. Dificultad para identificar los requerimientos.</p>	<p>El cumplimiento de los objetivos del proyecto se retrasa.</p>	<p>Realizar reuniones con los stakeholders para identificar lo más precisamente posible los requerimientos.</p>
<p>8. Aparición de problemas con las herramientas de desarrollo.</p>		<p>Realizar el sistema con herramientas conocidas ó de fácil aprendizaje.</p>
<p>9. Dificultad para</p>		<p>Buscar asesoría</p>

comprender y aplicar los principios pedagógicos necesarios.		especializada en el tema para que oriente al equipo de desarrollo así como realizar investigación propia.
---	--	---

Tabla D7. Manejo de Riesgos. Fuente: Elaboración propia.