Universidad de las Ciencias Informáticas

FACULTAD 6

Título: “Componente para la gestión de fórmulas de indicadores en SIGE”

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

Autor:

Rolando Morales Aguila

Tutores:

Ing. Héctor Luis Reyes

Ing. Adrián Rosales Cruz

La Habana, junio de 2013

“Año 55 de la Revolución”

“No sé cómo será la III Guerra Mundial, pero sí la IV… con piedras y palos.”

Albert Einstein

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis: ***"Componente para la gestión de fórmulas de indicadores en SIGE"*** y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Rolando Morales Aguila

_____

Firma del Autor

Ing. Héctor Luis Reyes

_____

Firma del Tutor

Ing. Adrián Rosales Cruz

_____

Firma del Tutor

DATOS DE CONTACTO

Autor: Rolando Morales Aguila
Universidad de las Ciencias Informáticas
La Habana, Cuba
E-mail: raguila@estudiantes.uci.cu

Tutor: Ing. Héctor Luis Reyes
Universidad de las Ciencias Informáticas
La Habana, Cuba
E-mail: hlreyes@uci.cu

Tutor: Ing. Adrián Rosales Cruz
Universidad de las Ciencias Informáticas
La Habana, Cuba
E-mail: adrianrc@uci.cu

A mi mamá y a mi padrastro por haberme apoyado estos cinco años.

A mi familia.

A Dainé por su ayuda y apoyo.

A René, Yunet y Graciela por aclarar todas mis dudas.

A mis amigos del grupo.

Y a mis tutores Héctor Luis y Adrián.

A mi abuela Taita.

A mi mamá por su amor incondicional como madre y padre.

A mi padrastro por haberme acogido como un hijo.

A la gente de mi barrio Puerto Arturo y a todos mis amigos.

RESUMEN

El Sistema Integrado de Gestión Estadística (SIGE) es el encargado de captar información estadística a través de formularios matriciales bajo convenio con la Oficina Nacional de Estadística e Información (ONEI). Este sistema requiere otro componente que facilite el trabajo con el mismo y eleve la productividad.

El objetivo del presente trabajo de diploma es realizar el análisis, diseño e implementación de un componente que contribuya al proceso de captación de información estadística. Como resultado se obtuvo la integración a SIGE de un componente para la gestión de fórmulas de indicadores.

Palabras claves: Sistema Integrado de Gestión Estadística (SIGE).

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÒN

Desde los comienzos de la civilización han existido formas sencillas de estadística, pues ya se utilizaban representaciones gráficas y otros símbolos en pieles, rocas, palos de madera y paredes de cuevas para contar el número de personas, animales o ciertas cosas. Hacia el año 3000 a.C. los babilonios usaban ya pequeñas tablillas de arcilla para recopilar datos en tablas sobre la producción agrícola y de los géneros vendidos o cambiados (1).

En un principio la estadística estuvo asociada a los Estados para ser utilizados por el gobierno. Así pues, los datos estadísticos se referían originalmente a los datos demográficos de una ciudad o estado determinados. Hoy el uso de la estadística se ha extendido más allá de sus orígenes. En nuestros días, la estadística se ha convertido en un método efectivo para entender datos y tomar decisiones en economía, política, ciencia sociales, psicológica, biología, física y otras ramas, y sirve como herramienta para relacionar y analizar dichos datos. El trabajo de la estadística no consiste ya sólo en reunir y tabular los datos, sino sobre todo en el proceso de interpretación de esa información (1).

El desarrollo de la información y el aumento de datos estadísticos en Cuba exigieron la utilización de medios de procesamiento automatizado para la obtención de los resultados finales de estadísticas con mayor precisión y menor período de tiempo. Sobre esta base se crea la actual Oficina Nacional de Estadísticas e Información (ONEI) mediante el Artículo 3 del Decreto Ley No.147 de fecha 21/04/94. La ONEI es una institución estatal que tiene la tarea de recopilar todo tipo de datos generados por las otras empresas, entidades e instituciones del país referentes a producción, consumo, salud, educación, salarios, etc. La misma controla, prácticamente, toda la información que pueda generar estadísticas en el país. Su principal misión es garantizar la producción de estadísticas de calidad a través del Sistema Estadístico Nacional, ejerciendo una adecuada dirección, ejecución y control de la captación de las cifras económicas y sociales, así como su difusión de acuerdo con los requerimientos de la economía y las demás necesidades del país en información estadística (2).

Para cumplir con su objetivo la ONEI ha introducido las más novedosas técnicas de almacenamiento y procesamiento de datos, esto ha sido posible gracias al trabajo en conjunto de la ONEI con la Universidad de las Ciencias Informáticas (UCI). Del trabajo entre estas dos instituciones nace el Sistema Integrado de Gestión Estadística (SIGE), desarrollado por el Centro de Tecnologías de Gestión de Datos (DATEC) perteneciente a la facultad 6.

SIGE tiene como objetivo principal captar la información estadística a nivel empresarial a través de formularios matriciales. Los formularios matriciales se definen a partir de indicadores (Herramientas para clasificar y definir objetivos e impactos) y aspectos (Elementos relacionados a los indicadores que enmarcan la dimensión de estos, entiéndase características más específicas). En el momento de captar la información el usuario tiene que dar valor a cada uno de estos indicadores. En muchos casos los valores de los indicadores deben ser calculados utilizando fórmulas matemáticas, proceso que se hace de forma manual trayendo consigo, un mayor consumo de tiempo, deshumanización de la tarea y un aumento considerable en el margen de error al calcular estos indicadores.

Debido a la necesidad de dar solución a la situación planteada se identifica como **problema de la investigación:** ¿Cómo contribuir a la mejora del proceso de captación de indicadores en el Sistema Integrado de Gestión Estadística?

La investigación tiene como **objeto de estudio** los Sistemas de Gestión Estadística y como **campo de acción** la Gestión de Indicadores.

Como **objetivo general** se plantea: Desarrollar un componente de gestión de fórmulas de indicadores para el Sistema Integrado de Gestión Estadística.

Desglosado en los siguientes **objetivos específicos:**

1. Caracterizar el proceso de gestión de fórmulas de indicadores.
2. Realizar la modelación del componente para la gestión de fórmulas de indicadores utilizando la metodología para el desarrollo de software seleccionado.
3. Realizar la implementación de las funcionalidades definidas.
4. Realizar las pruebas funcionales correspondientes que demuestren el correcto funcionamiento del componente para la gestión de fórmulas de indicadores.

Para dar cumplimiento a los objetivos trazados, se plantean las siguientes **tareas de la investigación:**

1. Estudio de las tecnologías vinculadas a la gestión de la información existentes en la actualidad.
2. Estudio de las formas de gestión de fórmulas de indicadores.
3. Selección de las herramientas idóneas y metodología de desarrollo que se utilizarán en el desarrollo del componente.
4. Identificación de los requisitos funcionales y no funcionales.

5. Modificación del diseño de base de datos de SIGE para que soporte la gestión de fórmulas de indicadores.
6. Análisis y diseño del componente para la gestión de fórmulas de indicadores.
7. Elaboración del modelo de implementación.
8. Diseño de los casos de prueba definidos a partir de los requisitos funcionales.
9. Realización y documentación de las pruebas funcionales.

Posibles resultados:

- Obtención de los artefactos del Análisis, Diseño, Implementación y Prueba del componente para la gestión de fórmulas de indicadores.

-Componente que permite la gestión de fórmulas de indicadores.

El presente trabajo está estructurado en tres capítulos:

-**Capítulo 1:** Fundamentación Teórica

En este capítulo se presenta la definición del marco teórico de la investigación. Se exponen los temas referentes a los sistemas estadísticos existentes en el mundo y en Cuba. Además, se aborda el estudio de la metodología, herramientas y tecnologías que serán utilizadas para el desarrollo del componente.

-**Capítulo 2:** Análisis y Diseño del Componente.

En este capítulo se describe de forma detallada cómo deberá funcionar el componente a desarrollar y las especificaciones de su implementación.

-**Capítulo 3:** Implementación y Prueba

En este capítulo se describe como fue implementado el componente, así como las garantías de su funcionamiento de acuerdo con lo previsto.

CAPITULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se presenta la definición del marco teórico de la investigación. Además, se aborda el estudio de la metodología, herramientas y tecnologías que serán utilizadas para el desarrollo del componente.

Para un mejor entendimiento del problema y de su solución se hace necesario el conocimiento de determinados **conceptos.**

1.1.1. Dato

Desde el punto de vista computacional se refiere a un conjunto de letras, números o signos que tengan al menos un significado, constituyendo de este modo una unidad básica de trabajo conformada por dos elementos: un atributo o nombre de dato y un valor o expresión particular de ese atributo(3).

1.1.2. Información

Es la estructuración de los datos de tal manera que permita aumentar el nivel de conocimiento de un fenómeno. Permite seleccionar elementos informativos o indicadores necesarios para formar una opinión sobre un determinado problema y así aumentar el conocimiento sobre un área de interés (4).

1.1.3. Sistemas de Información

Es el conjunto de personas, procedimientos y equipos diseñados y mantenidos para recoger, procesar, almacenar, analizar y utilizar la información (4).

1.1.4. Indicador

Es un punto de referencia que juega un papel descriptivo o un papel evaluativo. Es una señal que permite observar y medir el comportamiento de una determinada variable. (5)

Los indicadores deben cumplir una serie de características:

- Cuantificables

Normalmente se trata de un dato estadístico (porcentajes, tasas, razones...) que pretenden sintetizar la información que proporcionan los diversos parámetros o variables que afectan a la situación que se quiere analizar. No obstante, determinados indicadores que nos permiten caracterizar un territorio, sobre todo su singularidad cultural, no poseen carácter cuantificable.

- Precisos y procedentes de fuentes fiables

Ser fiables, en el sentido que cualquier cambio en el indicador se corresponda a un cambio en la variable que mida, y precisos para garantizar unos mínimos de rigurosidad en el estudio de la evolución temporal de los mismos.

- Disponibilidad periódica

Los indicadores se actualizan de forma periódica, lo que permite realizar comparaciones en el espacio y en el tiempo. Comparando el mismo indicador para el mismo grupo en varios momentos se podrán evaluar las evoluciones que han tenido lugar.

- Compatibilidad

Ser compatibles con otros indicadores, de manera que permitan la comparación y la interpretación de cambios de situación.

- Fácil comprensión

Ser comprendidos por los no especialistas. Lo que no significa que las técnicas de elaboración tengan que ser necesariamente simples, sino que deben ser presentadas de manera que sean fácilmente interpretadas.

Existen dos formas de agregarle a SIGE la funcionalidad de gestión de fórmulas de indicadores. Una es la integración de un componente existente y otra es el desarrollo de uno nuevo. A continuación se analizan opciones para la decisión sobre una de las 2 variantes.

En la actualidad existen gran cantidad de sistemas de gestión estadística, pero solo se estudiarán algunos de ellos, buscando si alguno puede o tiene un componente que gestione fórmulas de indicadores.

1.2.1 MicroSet NT

Es un sistema desarrollado en MS-DOS en los años 70 por informáticos de la ONEI. Posee parte de las funcionalidades demandadas por la ONEI pero no las contempla todas. Dado que se desarrolló como aplicación para consola (con elementos visuales para el modo texto), no es todo lo amigable que se puede esperar de una aplicación actual. MicroSet NT no evolucionó como aplicación para Windows y actualmente presenta problemas para ejecutarse sobre cualquier Windows que no pertenezca a la familia 9x.

Este sistema no es integrable con SIGE pero fue analizado para verificar si contaba con alguna forma de gestión de indicadores pero sus funcionalidades se limitan a digitar indicadores.

1.2.2. Census and Survey Processing System (CSPro)

CENSUS AND SURVEY PROCESSING SYSTEM es un sistema de gestión de información desarrollado por la oficina del Censo de los Estados Unidos. Es un sistema para el procesamiento de censos y encuestas, fácil de usar e implementado en plataforma Windows, es un paquete de software para la entrada, corrección, tabulación, diseminación de censos y datos de encuestas (6).

Después de haberse realizado el análisis de esta herramienta no encontramos ningún componente en ella que gestionara fórmulas de indicadores, por tanto quedo desechada la opción de utilizar esta herramienta o cualquiera de los componentes de la misma.

1.2.3. Sistema Integrado para la Gestión Estadística (SIGE)

SIGE fue desarrollado en conjunto con la ONEI de forma tal que contempla el marco metodológico y operativo de esta. Como sistema está integrado por módulos altamente especializados y cohesivos que permiten la gestión de los centros informantes, la elaboración del SEN (Sistema Estadístico Nacional) para realizar la captación de los datos estadísticos, la digitalización y validación de la información recolectada, y su posterior consulta y análisis, a tales efectos está integrado por los módulos de Registros y Clasificadores, Generador de Modelos y Encuestas, Entrada de Datos y Reportes respectivamente. SIGE tiene como objetivo principal captar la información estadística a nivel empresarial a partir de formularios matriciales. Los formularios matriciales se definen a partir de indicadores y aspectos (Fig.1). SIGE como política se orienta a estándares abiertos lo cual concuerda con la política del país hacia una soberanía tecnológica (7).

Entre los indicadores que conforman un formulario matricial siempre existe una relación. En algunos casos, esta relación puede ser matemática. Por ejemplo, en la Fig.1 se puede apreciar que existe un indicador denominado SUMA DE CONTROL que debe contener la sumatoria de los indicadores anteriores. En estos momentos SIGE no realiza ese cálculo automáticamente sino que debe ser introducido manualmente. Si se tiene en cuenta las miles de empresas existentes en el país que generan mensualmente decenas de formularios estadísticos, se estaría hablando entonces de digitar por parte de las Oficinas Municipales de Estadística e Información millones de indicadores cada mes, muchos de los

cuales podrían ser calculados automáticamente lo que representaría un ahorro de tiempo, mayor humanización del trabajo y reducción de los márgenes de error.

Página 1					
Título del Indicador	UM	Cód. Fila	AÑO ACTUAL		AÑO ANTERIOR
			Plan	Real	
Indicadores de Producción e Ingresos					
Producción Mercantil	MP	100			
Ingreso turístico	MCUC	200			
Ingreso de la Unidad Presupuestada (UP)	MP	300			
Ventas netas de bienes y servicios	MP	400			
Ventas netas de bienes y servicios: en divisa	MCUC	410			
Ventas netas de bienes y servicios: Producciones	MP	420			
Ventas netas de bienes y servicios: Producciones en divisa	MCUC	421			
Ventas netas minoristas	MP	430			
Ventas netas minoristas: en divisa	MCUC	431			
Ventas netas de gastronomía: en divisa	MCUC	441			
Ventas netas mayoristas	MP	450			
Ventas netas mayoristas: en divisa	MCUC	451			
Ventas netas de gastronomía	MP	440			
Ingresos por servicios a las personas	MP	460			
Ingresos por servicios a las personas: en divisa	MCUC	461			
Ventas en comedores y merenderos	MP	1400			
Indicadores del sector externo					
Exportaciones de servicios	MCUC	1500			
Importaciones de servicios	MCUC	1600			
SUMA DE CONTROL	Total	99999999			

Ilustración 1: Formulario de captura de datos estadísticos en SIGE

Si se quisiera desarrollar un nuevo componente para integrar a SIGE el mismo debería cumplir con diferentes requisitos. Funcionalmente debe ser capaz de permitir la gestión de fórmulas de indicadores. Respecto a la tecnología debe ser integrable a la arquitectura del sistema, por lo tanto debe ser un componente web, basado en tecnologías libres y además implementado con los lenguajes Javascript y PHP.

Cumplir con los requisitos anteriormente expuestos implica un reto y un riesgo porque se quiere agregar un nuevo comportamiento a un sistema que ya es funcional, por lo tanto requiere, además de la inclusión de nuevos componentes, la modificación de otros, que garantizaría el acoplamiento del nuevo desarrollado en el sistema.

Entre estas modificaciones algunas saltan a la vista en un análisis preliminar:

- Al diseño de base de datos para garantizar la persistencia de las fórmulas que se asociarán a los indicadores.

- A la vista del sistema para incluir en el proceso de diseño de formularios la gestión de fórmulas de indicadores.
- Al negocio para permitir el procesamiento de la nueva funcionalidad del lado del servidor.
- A los Objetos de Transferencia de Datos (DTO según siglas en inglés) que incluirán nueva información dentro de la descripción de los indicadores.

Resulta evidente entonces la necesidad de incluirle la nueva funcionalidad a SIGE, teniendo en cuenta las implicaciones asociadas al hecho de que es un sistema ya funcional.

1.3 Metodología, tecnologías y herramientas empleadas en la solución

Para el desarrollo del componente se hace necesario estudiar las metodologías, herramientas y tecnologías definidas por el grupo de arquitectura del Departamento de Soluciones Integrales de DATEC. Este ambiente se ajusta a la necesidad de migrar progresivamente hacia plataformas y aplicaciones de código abierto, en concordancia con el desarrollo del proceso de informatización de la sociedad como parte de la ejecución por el país de una política orientada a alcanzar la seguridad, invulnerabilidad e independencia tecnológica.

1.3.1 Metodología de Desarrollo de Software

Una metodología de desarrollo de software define un conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentación, y aspectos de formación para los desarrolladores de sistemas de información. Por tanto una metodología de desarrollo de software es un conjunto de componentes que especifican (8):

- Cómo se debe dividir un proyecto en etapas.
- Qué tareas se llevan a cabo en cada etapa.
- Qué salidas se producen y cuando se deben producir.
- Qué restricciones se aplican.
- Qué herramientas se van a utilizar.
- Cómo se gestiona y controla un proyecto.

1.3.2 Proceso Unificado Abierto (*Open Unified Process*, OpenUP)

Es un proceso modelo y extensible, dirigido a gestión y desarrollo de proyectos de software basados en desarrollo iterativo, ágil e incremental apropiado para proyectos pequeños y de bajos recursos; y es aplicable a un conjunto amplio de plataformas y aplicaciones de desarrollo. (9)

Se divide en cuatro fases:

1. **Concepción:** Primera de las 4 fases en el proyecto del ciclo de vida, acerca del entendimiento del propósito y objetivos y obteniendo suficiente información para confirmar que el proyecto debe hacer. El objetivo de ésta fase es capturar las necesidades de los stakeholder en los objetivos del ciclo de vida para el proyecto.
2. **Elaboración:** Es la segunda de las 4 fases del ciclo de vida del OpenUP donde se trata los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base de la arquitectura del sistema.
3. **Construcción:** Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta fase es completar el desarrollo del sistema basado en la Arquitectura definida.
4. **Transición:** Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, y evalúa la funcionalidad y performance del último entregable de la fase de construcción.

Propone 6 flujos de trabajo:

1. **Requerimientos:** Se realizan entrevistas con el cliente para comprender el problema a resolver y se definen los requerimientos.
2. **Análisis y Diseño:** Se realiza el diseño de los requisitos que serán después implementados.
3. **Implementación:** Se realiza la implementación del sistema basándose en el diseño realizado.
4. **Prueba:** Busca los defectos a los largo del ciclo de vida.
5. **Gestión del Proyecto:** Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
6. **Gestión de Configuración y Cambios:** Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización y actualización, control de versiones, etcétera.

Beneficios de su uso:

- Es apropiado para proyectos pequeños y de bajos recursos, permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito.
- Permite detectar errores tempranos a través de un ciclo iterativo.
- Evita la elaboración de documentación, diagramas e iteraciones innecesarias requeridas en la metodología RUP.
- Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas.

Por sus características y beneficios OpenUP es la metodología adecuada para el desarrollo del componente, ya que se cuenta con solo cuatro meses para el desarrollo del mismo con un grupo de desarrollo muy pequeño. Otra de las razones por la que se elige esta metodología es que permite disminuir las probabilidades de fracaso e incrementar las probabilidades de éxito. Además, detecta errores en fechas tempranas del desarrollo a través de un ciclo iterativo, evitando la elaboración de documentos, diagramas e iteraciones innecesarias.

1.3.3 Lenguaje de Modelado UML

UML (*Unified Modeling Language*): es un lenguaje de modelado de sistemas de software. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un plano del sistema, incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

Es importante resaltar que UML es un lenguaje para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software, pero no especifica en sí mismo qué metodología o proceso usar. (10)

Por parte del equipo de arquitectura del departamento de soluciones integrales se decide utilizar el lenguaje unificado de modelado UML-2.0.

1.3.4 Herramienta CASE

Las herramientas CASE (Ingeniería de Software Asistida por Computadora) son aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas ayudan en todos los aspectos del ciclo de vida de desarrollo del software, en tareas como el proceso de realizar un diseño del proyecto, cálculo de costos, implementación de parte del código automáticamente con el diseño.

Por parte del equipo de arquitectura se decide las utilizar Visual Paradigm 6.1 como herramienta de modelado para trabajar en el desarrollo del componente.

1.3.5 Tecnologías Web

Se describen a continuación todas las tecnologías web que van a ser utilizadas para el desarrollo del componente que se encuentran dentro de las políticas del grupo e desarrollo.

1.3.6 Javascript

Javascript es un lenguaje interpretado, al igual que Visual Basic, sin embargo, posee una característica que lo hace especialmente idóneo para trabajar en Web, ya que son los navegadores que utilizamos para viajar por ella los que interpretan los programas escritos en Javascript. De esta forma, podemos enviar documentos a través de la Web que llevan incorporados el código fuente de programas, convirtiéndose de esta forma en documentos dinámicos, y dejando de ser simples fuentes de información estáticas. (11)

1.3.7 Lenguaje de programación (PHP5)

PHP es un lenguaje de programación del lado del servidor, posee una gran librería de funciones y mucha documentación. Las ventajas de sus usos son las siguientes: es un lenguaje muy rápido, es de fácil aprendizaje, soporta la programación orientada a objetos y puede ser utilizado tanto sobre Linux como Windows. Puede conectarse con la mayoría de los gestores de bases de datos tales como: MySQL, PostgreSQL, Oracle, MS SQL Server, entre otros. Posee documentación en su página oficial la cual incluye la descripción y ejemplos de cada una de sus funciones. Es libre y no requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Se define por el equipo de arquitectura del departamento la utilización del lenguaje de programación Javascript para la manipulación de los eventos y los elementos HTML del lado del cliente, mientras que la

selección de PHP5 permite codificar la lógica de negocio del sistema que se encuentra del lado del servidor.

Framework

"Un framework es una aplicación reutilizable, semi-completa que puede ser especializada para producir aplicaciones concretas y específicas. El framework describe los objetos que componen el sistema, cómo éstos interactúan, y cuáles son sus responsabilidades."
[R. Johnson y B. Foote (Journal of Object-Oriented Programming – 1988)]

En la actualidad son muy usados para simplificar y agilizar el proceso de desarrollo de aplicaciones. Proporcionan un conjunto de librerías con funcionalidades que aumentan la facilidad del trabajo y disminuyen su complejidad. Además permiten reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.

1.3.8 Symfony

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. Symfony está desarrollado completamente con PHP5. Es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. (12)

1.3.9 ExtJS 2.2

ExtJS es una biblioteca JavaScript para el desarrollo de aplicaciones web enriquecidas, es de alto rendimiento y completamente orientada a objetos. Es compatible con la mayoría de los navegadores, hace uso de diferentes tecnologías como Ajax, DHTML y DOM. Brinda múltiples posibilidades para el trabajo con las validaciones, manejo de errores en el cliente y permite la personalización de temas de estilos. Basa toda su funcionalidad en Javascript a través de librerías YUI, jQuery, o haciendo uso de la librería nativa, así en tiempo de ejecución carga y crea todos los objetos HTML a través del uso intenso de DOM.

A propuesta del grupo de arquitectura del departamento se decide utilizar los frameworks Symfony 1.1.7 y ExtJS 2.2 teniendo en cuenta las potencialidades que brindan para agilizar el desarrollo del componente. Para el caso de Symfony, se destaca el uso del patrón arquitectónico Modelo-Vista-Controlador que facilita la separación de estos elementos, lo que permite modificaciones en cualquiera de las capas sin afectar a las demás. ExtJS por su parte tiene una base de componentes amplia que permite la reutilización y extensión de los mismos, así como también una vasta documentación y comunidad.

1.3.10 Sistema Gestor de Base de Datos (SGBD)

Es un software o conjunto de programas que permite crear y mantener una base de datos. El SGBD actúa como interfaz entre los programas de aplicación (usuarios) y el sistema operativo. Proporciona a los usuarios una visión abstracta de la información, es decir el sistema ahorra al usuario la necesidad de conocer los detalles de cómo se almacenan los datos. El objetivo principal es proporcionar un entorno eficiente a la hora de almacenar y recuperar la información de las bases datos. Facilita el proceso de definir, construir, y manipular bases de datos para diversas aplicaciones. (13)

1.3.11 PostgreSQL 8.4

PostgreSQL es un sistema gestor de base de datos objeto-relacional, bajo licencia BSD. La licencia BSD al contrario que la GPL permite el uso del código fuente en software no libre. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Se ejecuta en los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows, etc. Posee una documentación muy bien organizada, pública y libre. Tiene soporte nativo para los lenguajes: PHP, C, C++, Perl, Python, etc. Es altamente adaptable a las necesidades del cliente (14).

Características:

Atomicidad (Indivisible): es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.

Consistencia: es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la base de datos.

Aislamiento: es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generarán ningún tipo de error.

Durabilidad: es la propiedad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema (14).

A propuesta del grupo de arquitectura del departamento se decide utilizar Postgresql-8.4 como sistema gestor de base de datos, ya que es el gestor utilizado por SIGE en todas sus versiones y el componente a desarrollar debe regirse por la arquitectura de SIGE.

1.3.12 Servidor Web

Un servidor Web es un equipo que ejecuta el protocolo TCP/IP que devuelve páginas web a clientes que la solicitan. El protocolo usado para esta transferencia de informaciones HTTP (*Hyper Text Transfer Protocol*), que va encapsulado en TCP/IP. Así, un servidor web puede albergar sitios web para que sean consultables en todo el mundo por la red Internet (15).

1.3.13 Apache 2.0

El servidor HTTP Apache es potente y flexible. Es altamente configurable y extensible con módulos de terceros. Proporciona el código fuente completo, viene con una licencia sin restricciones y está en constante desarrollo. Puede ejecutarse en diferentes sistemas operativos como Windows y la mayoría de las versiones de Unix. Implementa muchas características de uso frecuente, tales como (16):

- Permite configurar fácilmente las páginas protegidas con contraseña con un enorme número de usuarios autorizados, sin empantanamiento del servidor.
- Respuestas personalizadas a los errores y problemas.
- Permite configurar los archivos o scripts CGI, incluso que sean devueltos por el servidor en respuesta a los errores y problemas, por ejemplo, un script de configuración para interceptar 500 errores de servidor.
- Permite al servidor distinguir entre las peticiones realizadas a diferentes direcciones IP o nombres (asignado a la misma máquina).

1.3.14 Entorno de Desarrollo Integrado (IDE)

Un entorno de desarrollo integrado o en inglés *Integrated Development Environment* (IDE) es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien, poder utilizarse para varios.

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Los IDEs pueden ser aplicaciones por si solas o pueden ser partes de aplicaciones existentes. (17)

1.3.15 NetBeans 7.0

NetBeans es un proyecto de código abierto dedicado a proveer productos de desarrollo de software que responden a las necesidades de los desarrolladores, los usuarios y las empresas que dependen de NetBeans como base para sus productos. Proporciona un entorno rápido de desarrollo integrado para crear, ejecutar y depurar aplicaciones PHP. Ofrece soporte completo, editor de HTML, Javascript y CSS. El editor reconoce código HTML en archivos Javascript y el código Javascript en los archivos HTML. El editor también reconoce HTML y Javascript en XHTML, PHP y archivos JSP. (18)

A propuesta del grupo de arquitectura del departamento se decide utilizar NetBeans 7.0, Apache 2.0 y PostgreSQL 8.4, teniendo en cuenta las potencialidades que brindan para agilizar el desarrollo del componente y que estas fueron las mismas que se definieron para la creación de SIGE.

A partir del estudio realizado de las tecnologías y herramientas, se analizaron las seleccionadas para el desarrollo del componente, ajustándose a las políticas definidas por el grupo de arquitectura del departamento. Se propone realizar un componente para la gestión de fórmulas de indicadores, donde este se integrará al sistema basándose en lo establecido, sin crear un conflicto entre herramientas, tecnologías y guiado por el buen camino de la soberanía tecnológica a través de la utilización del software libre.

Conclusiones del Capítulo

El análisis de diferentes sistemas estadísticos demostró la necesidad de un componente para la gestión de fórmulas de indicadores en el Sistema Integrado de Gestión Estadística. El estudio de las herramientas y tecnologías definió la línea base de la arquitectura: OpenUP como metodología de desarrollo de software, ExtJS en su versión 2.2 como framework de trabajo en el lado del cliente, Symfony en su versión 1.1.7 como framework de trabajo del lado del servidor y como SGBD PostgreSQL en su versión 8.4.

CAPITULO 2: ANÁLISIS Y DISEÑO DEL COMPONENTE

Introducción

En el capítulo se describen los principales conceptos de entorno que serán objeto de análisis para la realización de la fase de Análisis y Diseño del componente, a partir del modelo de dominio. Se identifican los requisitos funcionales y no funcionales a tener en cuenta para la implementación del componente. Se identifican los actores, casos de uso y las relaciones existentes entre ellos. Lo anterior será la base para la implementación del componente de gestión de fórmulas de indicadores.

2.1 Propuesta de solución

Se propone la implementación de un componente que permitirá a los trabajadores de SIGE, crear y asociar una fórmula a un determinado indicador, en un determinado formulario. El componente permitirá que cada indicador se calcule utilizando la fórmula que le fue asignada, evitando que los usuarios realicen cálculos complejos que puedan dar lugar a la ocurrencia de errores en tareas que pueden ser ejecutadas por los ordenadores. El proceso de creación de la fórmula será intuitivo y fácil para los usuarios, permitiendo que los mismos se familiaricen rápidamente con el componente y aumenten su productividad al trabajar con el mismo.

2.2 Modelo del Dominio

El modelo de dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción. El modelo de dominio muestra uno o más diagramas de clases que no contienen conceptos propios de un sistema de software sino de la realidad. Es un subconjunto del modelo de negocio y se realiza cuando no están claros los procesos o cuando no se identifican claramente los actores y trabajadores del negocio. Además, el modelo de dominio captura los tipos más importantes de objetos que existen, o los eventos que suceden en el entorno donde estará el sistema (19).

2.2.1 Diagrama conceptual del Modelo de Dominio

No están bien definidos los procesos del negocio por lo que se realiza el modelo de dominio, el cual tiene como objetivo fundamental comprender y describir los conceptos más importantes dentro del contexto del sistema, es una representación visual del entorno real del proyecto.

El modelo de dominio ocupa un rol protagónico en el desarrollo de software y además puede ser tomado como punto de partida para el diseño del sistema. El siguiente modelo de dominio tiene como objetivo contribuir a la comprensión del componente para la gestión de fórmulas de indicadores.

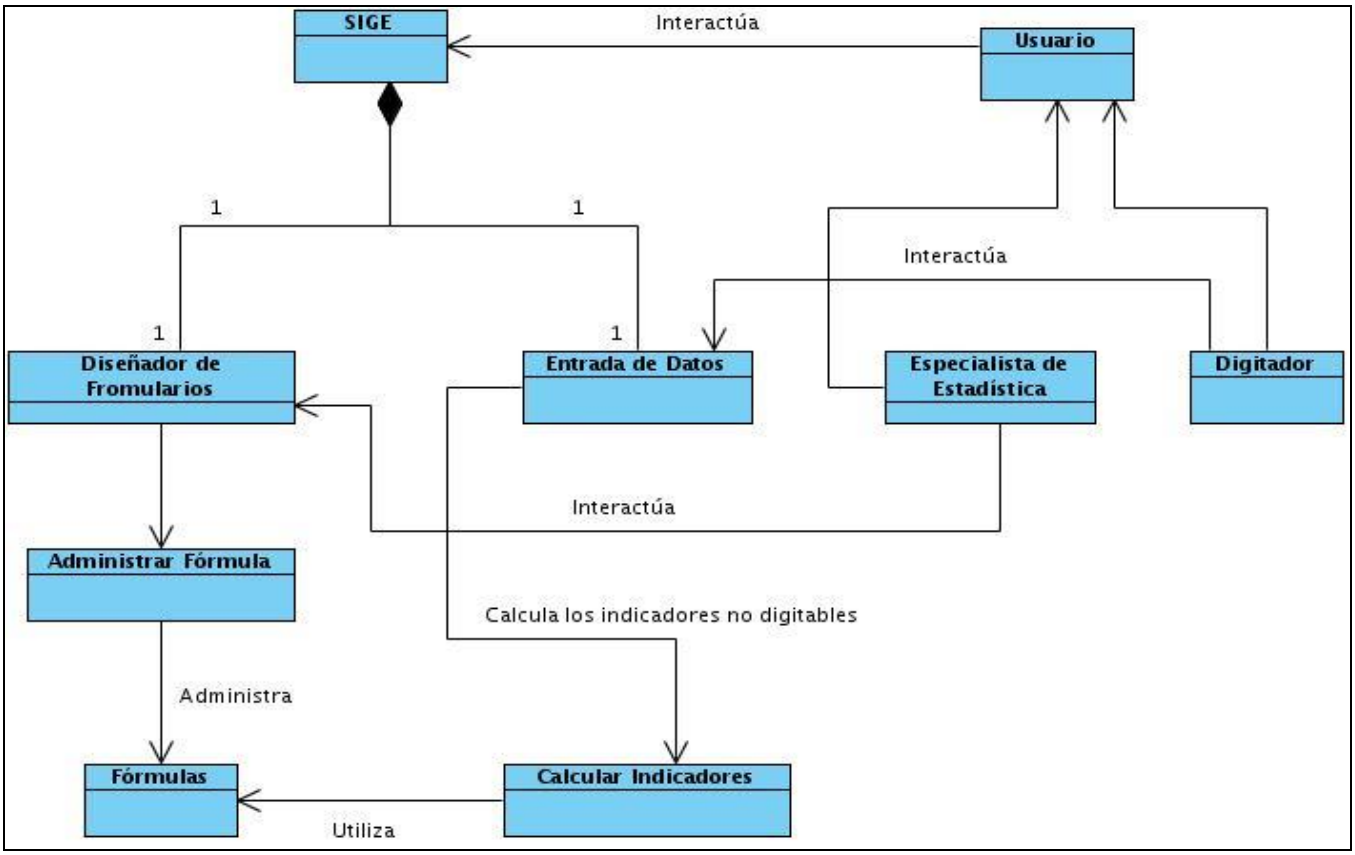


Ilustración 2: Diagrama del Modelo de Dominio

Definición de clases del modelo del dominio

SIGE: Sistema Integrado de Gestión Estadística.

Usuario: Persona que trabaja con el sistema.

Especialista de Estadística: Profesional encargado de diseñar los formularios.

Digitador: Profesional encargado de llenar los formularios de captura de datos.

Entrada de datos: Módulo encargado de la captura de datos.

Administrar Fórmula: Interfaz en la cual se gestionan las fórmulas.

Calcular Indicadores: Función que ejecuta el sistema para calcular aquellos indicadores que tengan asociada un fórmula.

Fórmulas: Tabla de la base de datos que almacena toda la información referente a cada una de las fórmulas.

2.3 Especificación de los Requisitos del Sistema

Los requisitos de un sistema se definen como las descripciones de como el sistema debe comportarse, la información acerca del dominio de la aplicación, las restricciones operativas del sistema o las especificaciones de las propiedades o atributos del sistema. Con los requisitos se pretende averiguar qué es lo que el cliente quiere del sistema y entender cuáles son sus necesidades en términos de diseño (20).

Para este trabajo el artefacto de especificación de requisitos de software debe contener una lista detallada y completa de los requisitos que debe cumplir el componente, donde el nivel de detalle de los requisitos tiene que ser claro, preciso y consistente.

2.3.1 Requisitos Funcionales

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera que este debe reaccionar a determinadas entradas y de cómo se debe comportar en situaciones particulares. En algunos casos, los requisitos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema debe hacer (21). En el componente a desarrollar se identifican los siguientes requisitos funcionales:

RF1: Crear Fórmula.

Descripción: El componente debe ser capaz de permitir al usuario crear de manera eficiente y fácil las fórmulas.

Entradas: Operando y operadores (operadores básicos como suma, resta, división, multiplicación. Números y agrupadores.)

Salidas: Texto con la fórmula.

RF2: Listar Fórmula.

Descripción: El componente debe ser capaz de mostrar la fórmula del indicador en caso de que este tenga asociada una fórmula.

Entradas: Identificador del indicador al cual se desea asignar una fórmula.

Salidas: Texto con la fórmula.

RF3: Salvar Fórmula.

Descripción: El componente debe ser capaz de salvar la fórmula, para que en otro momento la misma sea guardada en la base de datos.

Entradas: Texto con la fórmula.

Salidas: Mensaje que informa si la acción se produjo correctamente.

RF4: Eliminar Fórmula.

Descripción: El componente debe ser capaz de eliminar la fórmula asociada a un indicador.

Entradas: Identificador del indicador al que se desea eliminar la fórmula.

Salidas: Se elimina el texto con la fórmula del DTO (Objeto de Transferencia de Datos).

RF5: Modificar Fórmula.

Descripción: El componente debe ser capaz de hacer modificaciones a las fórmulas que están asociadas a los indicadores.

Entradas: Identificador del indicador al cual se desea modificar la fórmula.

Salidas: Mensaje que informa si la acción se produjo correctamente.

RF6: Calcular indicadores.

Descripción: El sistema debe ser capaz de calcular aquellos indicadores que tengan asociada una fórmula a partir de la misma.

Entradas: Fórmula con la cual se va a calcular dicho indicador.

Salidas: Se guardan los valores de estos indicadores en la BD.

2.3.2 Requisitos No Funcionales

Los requisitos no funcionales, como su nombre sugiere, son aquellos requisitos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes, como fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, se definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema (21).

Los requisitos no funcionales son importantes para que los clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con toda la funcionalidad requerida, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

Para el componente propuesto se definen los siguientes requisitos no funcionales:

Restricciones del Diseño:

Para la implementación del sistema se requiere del uso de las siguientes herramientas:

RFN1: Lenguaje y marco de trabajo (framework) para el desarrollo del sistema del lado del servidor.

El componente deberá ser implementado en el lenguaje de programación PHP. Como framework de desarrollo se usará Symfony 1.1.7.

RFN2: Lenguaje y marco de trabajo (framework) para el desarrollo del sistema del lado del cliente.

Unas de las bibliotecas fundamentales en el desarrollo de la herramienta es la de ExtJS la cual permite el diseño de interfaces visuales interactivas usando metodologías como AJAX y permiten crear aplicaciones WEB con la apariencia de escritorio.

Interfaz:

RFN3: Interfaces de usuario.

Las interfaces de usuario serán diseñadas a modo de aplicaciones RIA (*Rich Internet Application*) lo que permite a los usuarios contar con aplicaciones web muy similares a las aplicaciones de escritorios. Para lograr este objetivo se usará la librería Javascript, ExtJS, la cual conjuga una serie de componentes visuales que proveen funcionalidades que ayudan al diseño de este tipo de aplicaciones WEB con apariencia de escritorio.

RFN4: Hardware.

Cliente

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 256MB.

Servidor

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 1GB.

RFN5: Software.

Un servidor en el que se instale únicamente SIGE con los siguientes elementos.

- Sistema operativo GNU/Linux Ubuntu 10.10 o versión superior.
- Servidor de aplicaciones Apache, versión 2.
- Lenguaje de programación PHP y librerías versión 5.
- Sistemas Gestor de Bases de Datos PostgreSQL, versión 8.4.

En los clientes Mozilla Firefox 3.4 o superior.

2.4 Modelo de Casos de Uso del sistema

El modelo de casos de uso describe un sistema en términos de sus distintas formas de utilización, cada una de las cuales se conoce como un caso de uso. Cada caso de uso o flujo se compone de una secuencia de eventos iniciada por el usuario. Dado que los casos de uso describen el sistema a desarrollar, los cambios en los requisitos significan cambios en los casos de uso (19).

El modelo de casos de uso está compuesto por actores, casos de uso y la relación que existe entre ellos. Representa un esquema que recoge las funcionalidades del sistema que se automatizan y determina el uso que tendrá desde el punto de vista del usuario, ya que su construcción es en base a las necesidades del mismo.

El mismo representa las relaciones existentes entre actores y casos de uso. Los actores son terceros fuera del sistema que interactúan con él (puede ser cualquier persona, individuo, entidad, organización, máquina o sistema de información externo; con los que el sistema interactúa) y los casos de uso son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores (7).

2.4.1 Definición de los Casos de Uso del Sistema

Un caso de uso es una manera de utilizar el sistema o de interactuar con él. Los casos de uso proporcionan una definición de las necesidades a cubrir por un proyecto desde el punto de vista del usuario. Por tanto es una técnica utilizada para ayudar al cliente a determinar sus necesidades y requisitos. Cada caso de uso es una recopilación de sucesos cuyo evento inicial lo provoca un actor o entidad externa, especificando la interacción que existe entre actor y sistema (22).

2.4.2 Definición de los actores del sistema

Los actores del sistema son entidades distintas a los usuarios en el sentido de que estos son las personas reales que utilizaran el sistema, mientras que los actores representan cierta función que una persona real realiza. Los actores modelan cualquier entidad externa que necesite intercambiar información con el sistema. No están restringidos a ser personas físicas, por lo que pueden representar otros sistemas externos al actual. Lo esencial es que los actores representan entidades externas al sistema. Además cada uno de estos actores podrá ejecutar una o más tareas del sistema (19).

Se identificaron los siguientes actores para el componente propuesto:

Actor	Descripción
Especialista de Estadística	El especialista de estadística tiene la responsabilidad de diseñar formularios y encuestas, es la persona que se encargada de definir cuales indicadores serán calculados, y en tal caso tiene la tarea de crear una fórmula para cada uno de estos indicadores.
Digitador	El digitador tiene la responsabilidad de captar información utilizando los formularios y encuestas anteriormente diseñados por el especialista de estadística.

Tabla 1: Actores del componente propuesto

2.4.3 Diagrama de Casos de Uso del Sistema

Los diagramas de casos de uso sirven para mostrar las funciones de un sistema de software desde el punto de vista de sus interacciones con el exterior y sin entrar ni en la descripción detallada ni en la implementación de estas funciones.

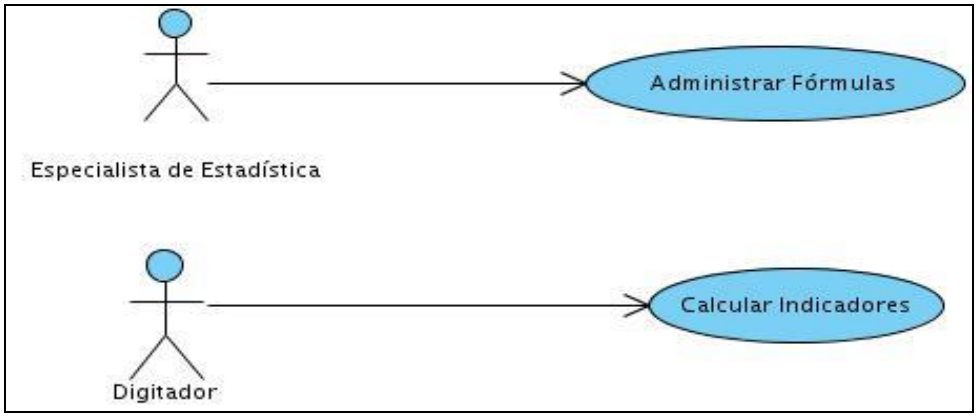


Ilustración 3: Diagrama de Casos de Uso del Sistema

2.4.4 Descripción textual de los Casos de Uso del Sistema

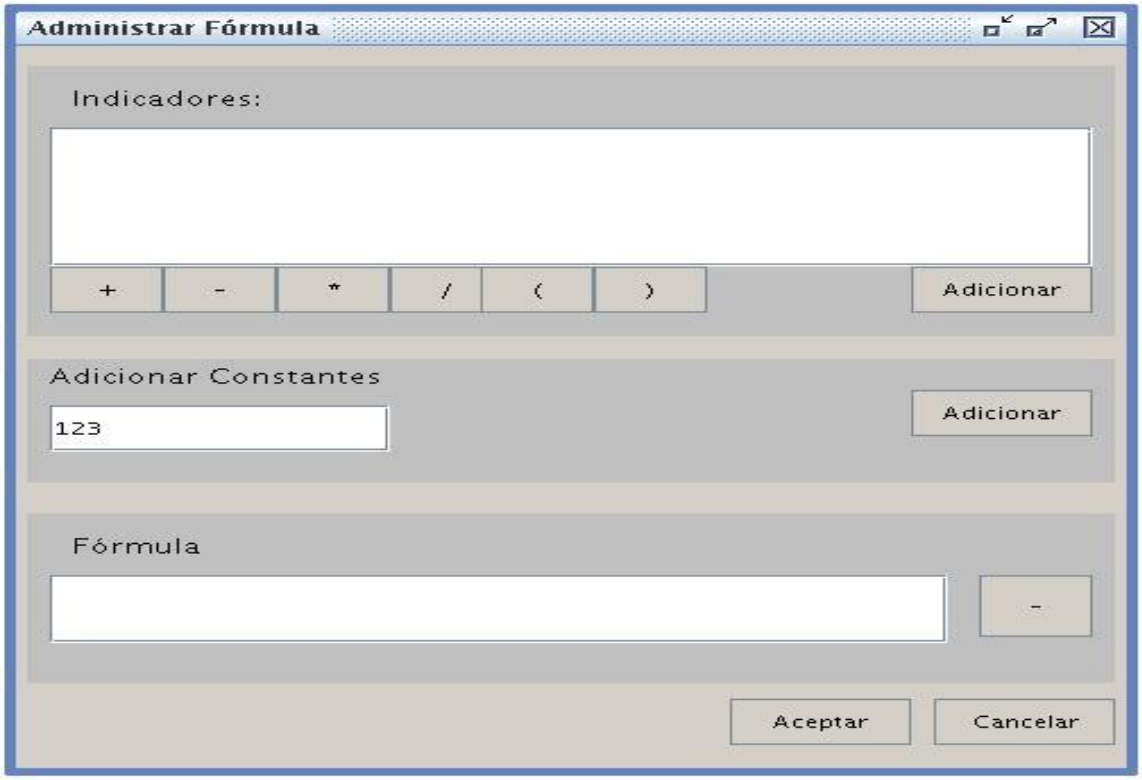
Cada caso de uso se detalla habitualmente mediante una descripción textual que describe la funcionalidad que se construirá en el componente propuesto. A continuación se muestra un ejemplo:

Caso de Uso:	Administrar Fórmula.

Actores:	Especialista de Estadística
Resumen:	Se administra todo lo referente a la fórmula de un indicador en una determinada página de un formulario, las acciones que se podrán ejecutar son: asignar una fórmula al indicador, listar, modificar y eliminar la fórmula del indicador en caso de que el mismo ya tuviera una fórmula asignada. Todas estas acciones se realizarán sobre un DTO (objeto de transferencia de datos) no directamente sobre la base de datos. Estas acciones solo se verán reflejadas en la Base de Datos cuando se guarde el formulario.
Precondiciones:	Para que se ejecute este caso de uso, el especialista de estadística tiene que haber ejecutado la acción de crear un nuevo formulario o la de modificar un formulario existente, y luego debe seleccionar el indicador al cual desea administrar la fórmula.
Referencias	**RF1, RF2, RF3, RF4, RF5,**
Prioridad	**Crítico**

Flujo Normal de Eventos	
Sección "Asignar Fórmula"	
Acción del Actor	**Respuesta del Sistema**
1- Selecciona un indicador y selecciona la opción Administrar Fórmula.	2-Se muestra la ventana Administrar Fórmula.
3-Crea la fórmula y selecciona la opción Aceptar	4-Valida que la fórmula esté formada correctamente. 5-Se guarda la fórmula en el DTO, se muestra un mensaje de que la fórmula se guardó correctamente y se cierra la ventana.

Prototipo de Interfaz



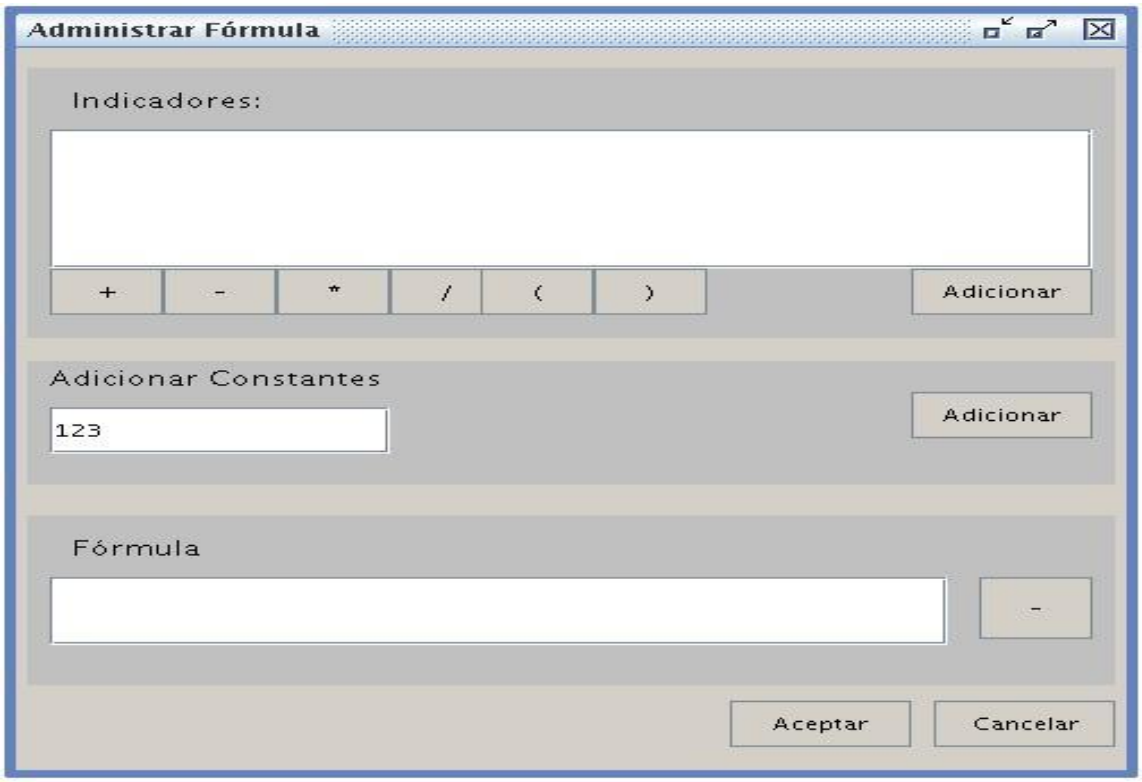
Flujos Alternos

Acción del Actor	**Respuesta del Sistema**
	5a-Se muestra un mensaje de error informando que la fórmula no fue creada correctamente y que debe modificar dicha fórmula para que pueda ser guardada, y se mantiene visible la ventana para volver al paso 3.

Prototipo de Interfaz

Sección “Modificar Fórmula”

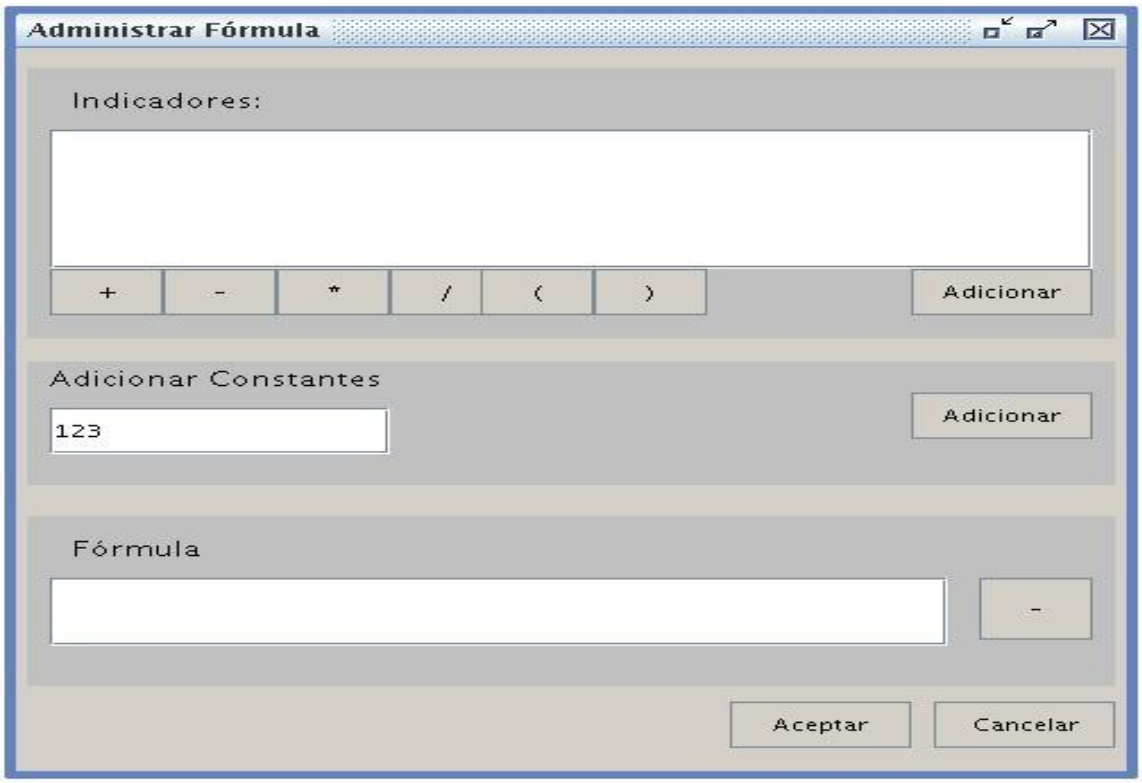
Acción del Actor	**Respuesta del Sistema**
1- Selecciona un indicador y selecciona la opción Administrar Fórmula.	2-Se muestra la ventana Administrar Fórmula, con la fórmula de dicho indicador.
3- Modifica la fórmula y selecciona la opción Aceptar	4-Se modifica fórmula en el DTO, se muestra un mensaje informando que la fórmula fue modificada satisfactoriamente y se cierra la ventana.
Prototipo de Interfaz	



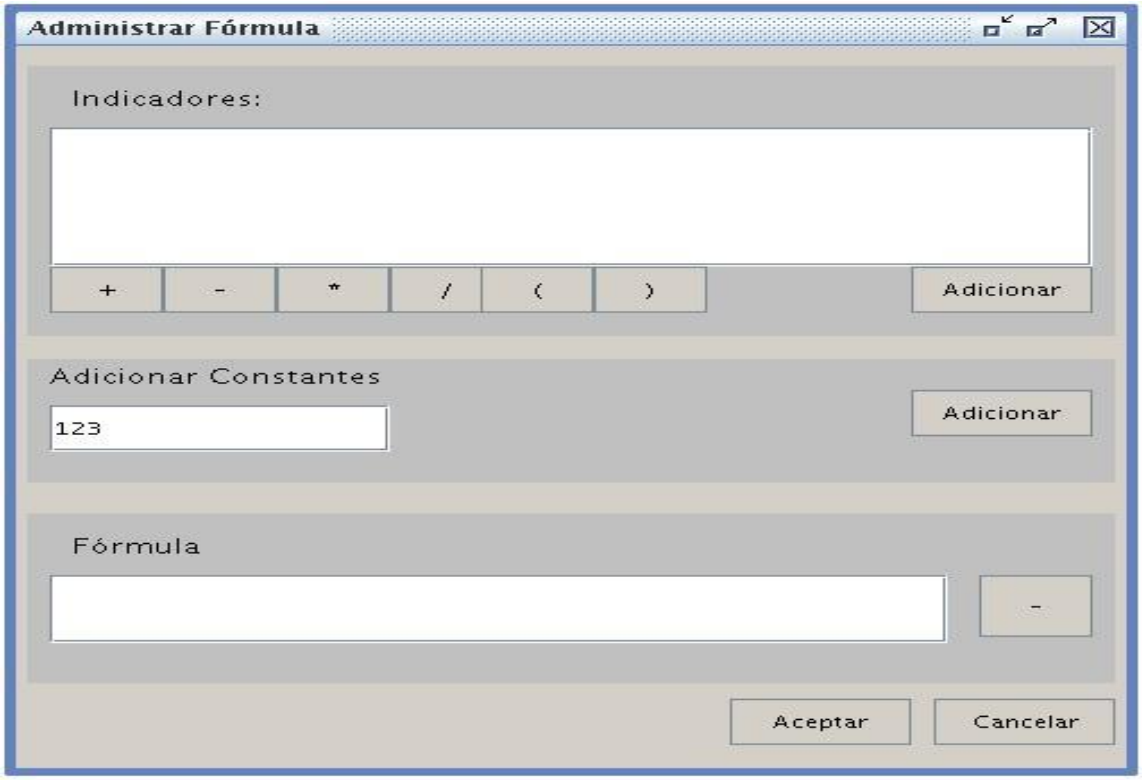
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4a- Se muestra un mensaje de error informando que

	la fórmula no fue modificada, y se mantiene visible la ventana para volver al paso 3.
Sección "Eliminar Fórmula"	
Acción del Actor	**Respuesta del Sistema**
1- Selecciona un indicador y selecciona la opción Administrar Fórmula.	2-Se muestra la ventana Administrar Fórmula, con la fórmula de dicho indicador.
3- Selecciona la opción Eliminar Fórmula.	4-Se elimina la fórmula del DTO, se muestra un mensaje informando que la acción se produjo satisfactoriamente y se cierra la ventana.

Prototipo de Interfaz



Sección "Listar Fórmula"	
Acción del Actor	**Respuesta del Sistema**
1-Selecciona un indicador y selecciona la opción Administrar Fórmula.	2-Se muestra la ventana Administrar Fórmula, con la fórmula de dicho indicador.
3- Selecciona la opción Aceptar y se cierra la ventana.	
Prototipo de Interfaz	



Poscondiciones	El indicador queda con una fórmula asignada en el DTO. El indicador queda sin una fórmula asignada en el DTO. El indicador queda con su fórmula modificada en el DTO.

Tabla 2: Descripción del caso de uso "Administrar de Fórmula"

2.5 Diseño del Sistema

El diseño del sistema define la arquitectura y estructura de hardware, software, componentes, módulos y datos de un sistema de cómputo. Se refiere a la formulación de especificaciones para el nuevo sistema o subsistema propuesto, de manera que satisfaga los requisitos determinados durante la fase de análisis, tanto desde el punto de vista funcional como del no funcional.

Para hacer un diseño eficiente se tomarán en cuenta un conjunto de patrones, ya que los mismos constituyen una guía para resolver problemas comunes en programación que generalmente encuentras en el diseño de un programa. Cada patrón explica cómo resolver un determinado problema, bajo determinadas circunstancias.

2.5.1 Patrones arquitectónicos y de diseño

Se emplea Symfony como framework de desarrollo el cual utiliza lo mejor de la arquitectura MVC e implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo representándolo a través de tres elementos fundamentales: el modelo, la vista y el controlador.

El **Modelo** representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. La base de datos pertenece a esta capa.

La **Vista** transforma el modelo en una página web que permite al usuario interactuar con ella. En Symfony la capa de la vista está formada principalmente por plantillas en PHP.

El **Controlador** se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista. Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista (12).

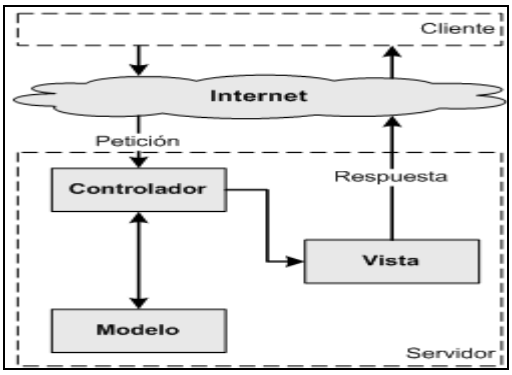


Ilustración 4: Patrón de Diseño MVC

Symfony está concebido de tal manera que obliga el uso de diferentes patrones de diseño. Los patrones de diseño empleados en la solución pertenecen al conjunto de patrones GRASP, la utilización de estos patrones ha ayudado a refinar el diseño y a asignar las responsabilidades de las distintas clases de diseño, haciéndolas más sencillas y reutilizables.

Patrones GRASP:

Experto: Plantea que se debe asignar la responsabilidad de realizar determinada operación, a la clase que tiene la información necesaria para cumplir con dicha responsabilidad. El experto es usado más que cualquier otro patrón en la asignación de responsabilidades, es un principio usado continuamente en el diseño orientado a objetos (25).

En la arquitectura de Symfony, específicamente en el modelo, existen dos tipos de clases fundamentales:

- Las clases que se encargan de la abstracción de datos que son las responsables de realizar todas las operaciones con la BD.
- Las clases de acceso a datos que son las responsables de interactuar con las clases de abstracción de datos, devuelven los objetos que necesitan los controladores en su forma original.

Symfony genera 4 clases por cada tabla de la BD, en el componente a realizar tendrá una tabla denominada Fórmula, por tanto se generarán las siguientes clases: Fórmula, “BaseFórmula”, “FórmulaPeer” y “BaseFórmulaPeer”. De estas cuatro clases, se percibe que las clases que trabajan directamente con la BD, son las terminadas en “Peer”, estas clases son las encargadas de hacer las consultas a la BD utilizando Propel, por tanto estas clases como clases de abstracción de datos son las que tienen los atributos necesarios para realizar dicha función, por tanto deben implementar la

responsabilidad de realizar las acciones directamente con la Base de Datos y aquí es donde se aplica el patrón Experto.

Alta Cohesión: Plantea que la información que almacena una clase debe de ser coherente y debe estar relacionada con la clase (25).
Symfony organiza el trabajo en cuanto a la estructura del proyecto, lo cual permite crear y trabajar con clases con una alta cohesión. Este patrón se evidencia en las clases del modelo, las cuales tienen solo los atributos y métodos necesarios para que estas cumplan con su función.

Bajo Acoplamiento: Plantea que tener las clases lo menos ligadas entre sí permite que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases (25).

En la capa del Modelo se encuentran las clases que implementan la lógica de negocio y de acceso a datos, estas clases tienen pocas asociaciones con otras de la Vista o el Controlador por lo que la dependencia en este caso es baja, poniéndose de manifiesto el patrón Bajo Acoplamiento.

Controlador: Consiste en tener una clase que sirva como intermediaria entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado (25).

Todas las peticiones Web son manejadas por un solo controlador frontal (sfAction), que es el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL entrada por el usuario.

2.5.2 Modelo de Diseño

Los modelos de diseño muestran los objetos o clases en un sistema y, donde sea apropiado los diferentes tipos de relaciones entre estas entidades. Los modelos de diseño son esencialmente el diseño mismo. Son el puente entre los requerimientos y la implementación del sistema. Tienen que ser abstractos con el fin de que el detalle innecesario no oculte las relaciones entre ellos y los requisitos del sistema. Sin embargo, también tienen que incluir suficiente detalle para que los programadores tomen las decisiones de implementación (23).

2.5.3 Diagramas de clases del diseño

El diagrama de clases del diseño es una representación concreta de lo que se debe implementar. Estos diagramas representan la parte estática del sistema a través de la representación de las clases y sus relaciones.

A partir de la descripción detallada de los casos de usos del sistema, se modelaron los diagramas de clases del diseño. A continuación se muestra un ejemplo:

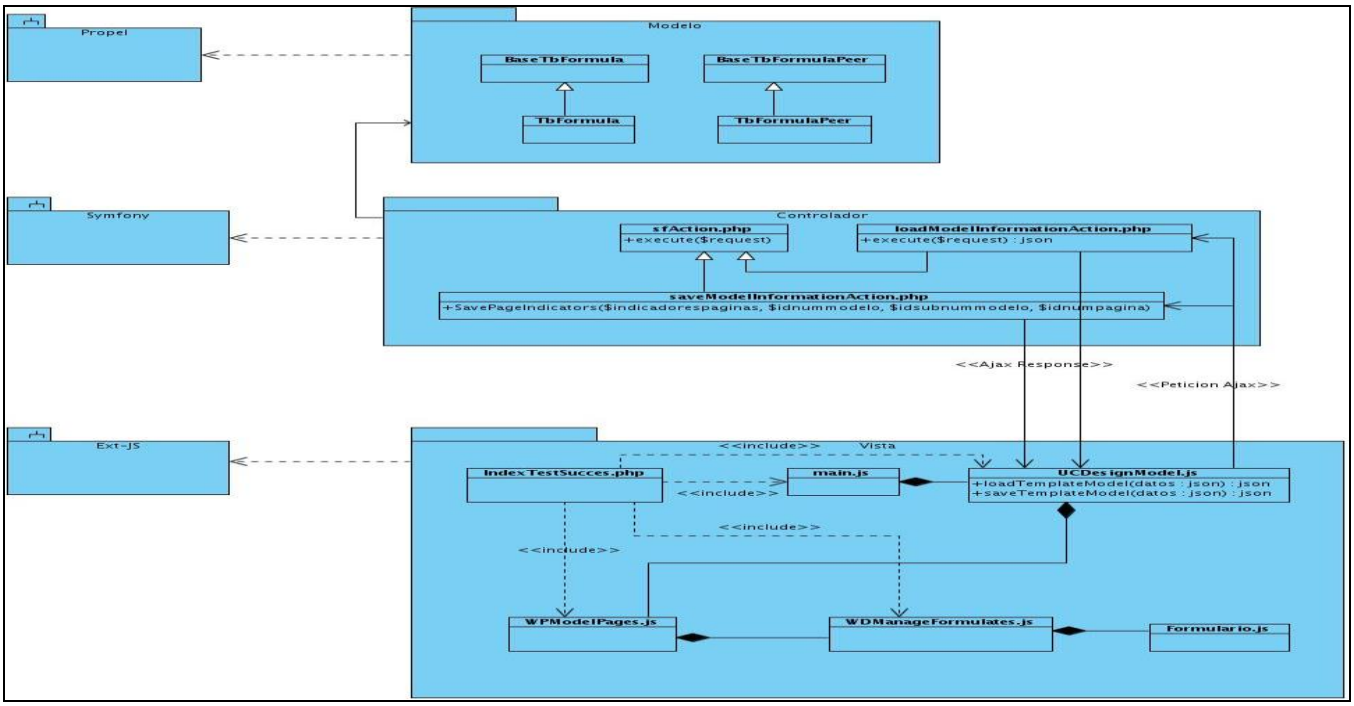


Ilustración 5: Diagrama de clases del diseño CU Administrar Fórmula

A continuación se muestran las clases identificadas en el componente:

- indexSucces.php
- main.js
- UCDesignModel.js
- WPModelPages.js
- WDManagesFromulates.js
- SaveModelInformationAction.php
- loadModelInformationAction.php

Se utiliza el modelo vista controlador uno de los patrones más utilizados en Symfony. El Modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. La Vista transforma el modelo en una página web que permite al usuario interactuar con ella y el Controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

Como se trata de un componente para una aplicación enriquecida para la web, es necesario considerar los lenguajes que se encontrarán del lado del servidor entre ellos está PHP. Las interfaces de red establecidas por cada módulo del sistema permiten al servidor que ejecute el servicio enrutamiento y acceso remoto para la comunicación. Los servicios en Symfony se definen como las acciones (clase SaveModelInformationAction.php y loadModelInformationAction.php) agrupada en el correspondiente módulo Symfony al que pertenece, se destaca la relación que existe de flujo de información entre el componente módulo del lado del servidor. En Symfony la capa del controlador, que contiene el código que une la capa de negocio con la de presentación, está dividida en varios componentes que se utilizan para diversos propósitos, aquí es donde aparece como primer elemento el controlador frontal, que es el único punto de entrada a la aplicación, carga la configuración y determina la acción a ejecutarse. Además esta capa cuenta con las acciones, las cuales contienen la lógica de la aplicación, verifica la integridad de las peticiones y preparan los datos requeridos por la capa de presentación, utilizando los componentes de Symfony. Luego de asociar la acción, esta se comunica con la capa donde radica el paquete de clases con sus métodos y funciones necesarios para responder las peticiones del usuario, donde esta nueva capa interactúa con el modelo para el trabajo con los datos.

El modelo es el encargado de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes del tipo de gestor de bases de datos utilizado por la aplicación.
Las clases de la capa del modelo se generan automáticamente, en función de la estructura de datos de la aplicación. La librería Propel se encarga de esta generación automática, ya que crea el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. Cada tabla del modelo de datos genera 4 clases pero las verdaderas clases presentan la estructura de Clase.php y ClasePeer.php.
Las relaciones fundamentales entre estos elementos son de dependencia y las de flujo de datos. La relación de flujo de datos entre la interface y el modelo representa la correspondencia de las invocaciones

del cliente con las acciones que del lado del servidor darán respuesta a las mismas, se transfiere tanto en la solicitud como en la respuesta el objeto con todos los atributos que se envían.

2.5.4 Diagramas de interacción del diseño

El diagrama de interacción muestra las interacciones entre objetos mediante transferencia de mensajes entre objetos o subsistemas, por lo que son empleados para modelar aspectos dinámicos del sistema. Los diagramas de colaboración y de secuencia son dos tipos de diagramas de interacción que son semánticamente equivalentes pero sin embargo los diagramas de colaboración destacan el orden estructural de los objetos que interactúan y los de secuencia destacan el orden temporal de los mensajes. Para la realización de los casos de uso del diseño es más factible el empleo de los diagramas de secuencia ya que representan con más claridad el flujo de las acciones que debe realizar el sistema (23).
A continuación se muestra un ejemplo:

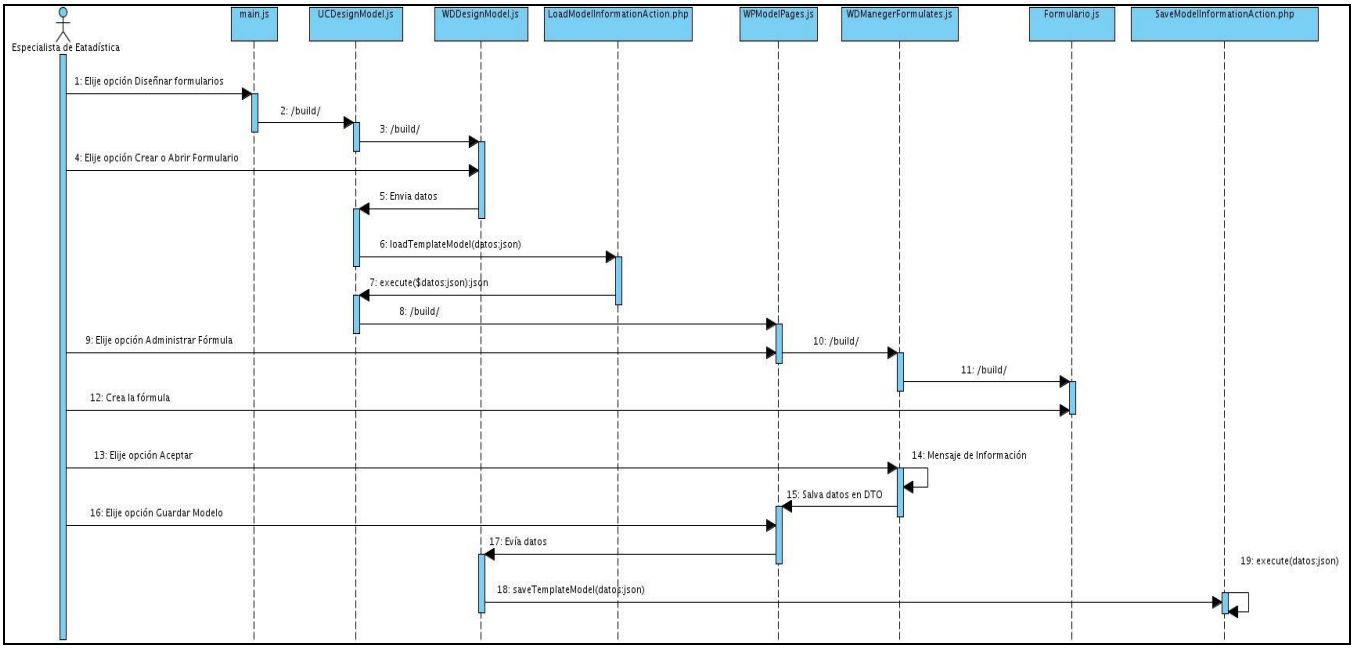


Ilustración 6: Diagrama de secuencia del escenario Asignar Fórmula del CU Administrar Fórmula

El caso de uso comienza cuando el usuario elije la opción “Diseñar Formularios”, el usuario puede elegir crear un nuevo formulario o abrir un formulario existente, se tienen que realizar transformaciones en la clase “loadModelInformationAction” para que cargue las fórmulas de aquellos indicadores que estén asociados con alguna de ellas. También ha de modificarse la clase “saveModelInformationAction” para

que guarde en la base de datos las fórmulas que fueron asignadas a determinados indicadores. Paso seguido el usuario elije en la vista “WPModelPages” el indicador al cual desea asignarle una fórmula y elije la opción “Administrar Fórmula”, acción que muestra la vista “WDManegerFormulates” en la cual el usuario crea la fórmula de dicho indicador. Después de creada la fórmula el usuario elije la opción “Aceptar” y el sistema valida que esté formada correctamente la fórmula y muestra un mensaje de información. En caso de que la fórmula haya sido creada correctamente la misma es salvada en el DTO (Objeto de Transferencia de Datos), en el momento que el usuario seleccione la opción “Guardar Modelo” el DTO es enviado a la clase “saveModelInformationAction” para que el modelo sea salvado en conjunto con las fórmulas.

2.5.5 Modelo de Datos

Una característica fundamental del enfoque de base de datos es que proporcionan un cierto nivel de abstracción de los datos, ocultando detalles acerca de su almacenamiento que no son necesarios conocer para la mayor parte de los usuarios. Un modelo de datos es la herramienta principal usada para proporcionar esa abstracción. El modelo de datos, en términos generales, proporciona un conjunto de conceptos que permiten modelar, principalmente, la estructura, un conjunto de operaciones básicas el comportamiento de toda la base de datos (24).

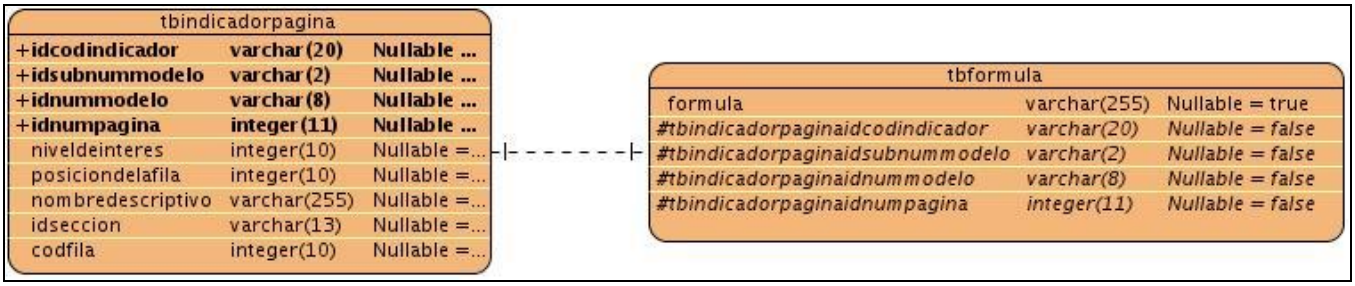


Ilustración 7: Diseño de la Base de Datos.

Para garantizar la persistencia de las fórmulas, se crea una tabla llamada “tbfomula” que formará parte de la Base de Datos de SIGE, la misma tiene relación con la tabla “indicadorespagina” garantizando que la fórmula se relacione con los indicadores solo cuando estos formen parte de un formulario y sea necesario asignar una fórmula a un determinado indicador.

2.5.6 Modelo de Despliegue

El modelo de despliegue presenta la arquitectura física del sistema por medio de los nodos Inter-conectados. Los nodos son los elementos hardware que van a soportar el software del sistema desarrollado, mostrando también las conexiones existentes entre estos nodos.

Del modelo de despliegue podemos observar que:

- Los nodos representan un recurso de cómputo, un procesador o un dispositivo de hardware similar.
- Los nodos poseen relaciones que representan medios de comunicación entre ellos, tales como HTTP, USB, TCP/IP, etc.
- Puede describir diferentes configuraciones de red, incluidas las configuraciones para prueba y simulación.

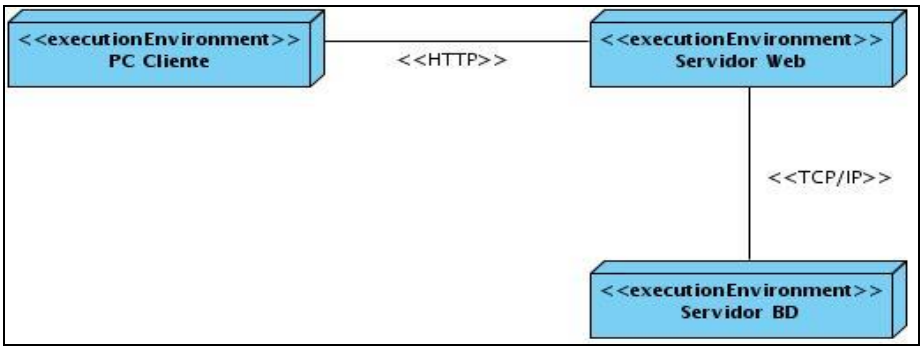


Ilustración 8: Modelo de Despliegue.

Estaciones de trabajo con la Aplicación Cliente

Estaciones de trabajo que el usuario utilizará para acceder al componente.

Servidor de Aplicación

Servidor de aplicación utilizado para la publicación de la aplicación; y para lograr la conexión del sistema con la PC Cliente se utiliza HTTP como protocolo de comunicación. Es la herramienta principal para ejecutar la lógica de negocio en el lado del servidor. Es el responsable de ejecutar el código de las páginas servidor. Se utiliza el servidor de aplicación Apache.

Protocolos de Comunicación

Un protocolo de comunicación es un conjunto de reglas establecidas entre dos dispositivos para permitir la comunicación entre ambos.

Conexión HTTP: Es el protocolo utilizado entre los navegadores de los clientes y el servidor Web. Este elemento de la arquitectura representa un tipo de comunicación no orientado a la conexión entre clientes y servidor.

Conexión TCP/IP: Es la base del Internet que sirve para enlazar computadoras. El protocolo TCP/IP es utilizado para establecer la conexión entre el servidor de aplicación y el servidor de base de datos.

Servidor de Base de Datos

Se refiere a un servidor que radica en cada nodo regional en el cual el cliente define que sean guardados los datos. En el servidor central estarán almacenados todos los datos recopilados por todos los nodos regionales.

Conclusiones del Capítulo

La realización del análisis del componente propuso una representación visual de las clases conceptuales del entorno a través del modelo de dominio. Se identificaron los actores del sistema, dos casos de usos, seis requisitos funcionales y cinco no funcionales que el sistema debe cumplir para su correcto funcionamiento. Se realizó el modelo de diseño con el propósito de describir cómo se debe implementar el sistema. Se utilizaron los diagramas de secuencia para inspeccionar los aspectos dinámicos del módulo y se identificaron los patrones de diseño a utilizar.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

Introducción

En este capítulo se desarrolla el flujo de trabajo de Implementación, se describen sus principales artefactos, destacando el Modelo de Implementación que incluye componentes, subsistemas de implementación y diagramas de componentes. Después de la implementación del componente a partir de las clases del diseño se realizan las pruebas necesarias al componente con el objetivo de obtener funcionalidades con la menor cantidad de errores posibles, aplicándole pruebas funcionales y de usabilidad.

3.1 Modelo de Implementación

El modelo de implementación describe cómo los elementos del modelo de diseño, como las clases, se implementan en términos de componentes, ficheros de código fuente, ejecutables, etc. Describe también cómo se organizan los componentes de acuerdo a los mecanismos de estructuración disponibles en el entorno de implementación y lenguajes de implementación empleados, y cómo dependen los componentes unos de otros. Esta descripción es de gran utilidad a la hora de implementar el sistema, facilita la organización del trabajo y lo hace más entendible a los desarrolladores (26).

3.1.1 Diagrama de componentes

Los diagramas de componentes describen cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros. Debido a que los diagramas de componentes son más parecidos a los diagramas de casos de usos, éstos son utilizados para modelar la vista estática y dinámica de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema (26).

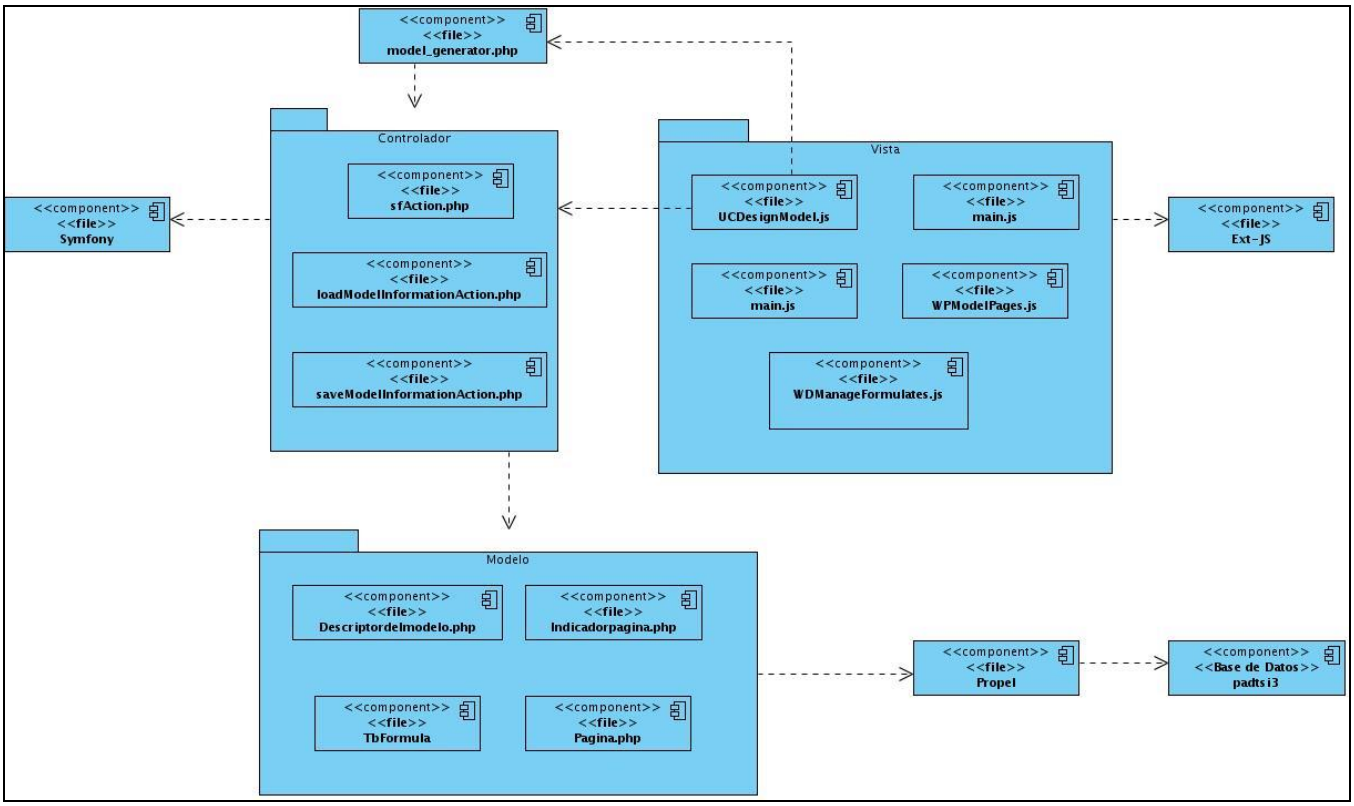


Ilustración 9: Diagrama de componentes CU Gestionar Fórmulas.

Symfony como framework de desarrollo que se emplea toma lo mejor de la arquitectura MVC e implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo representándolo a través de tres elementos fundamentales: el modelo, la vista y el controlador.

El diagrama de componentes realizado anteriormente representa una descripción de cómo se organizan los componentes y como dependen unos de otros. En la capa Vista se encuentra reflejado el paquete de componentes Vista, el cual lleva incluido el archivo que representará la interfaz de usuario de acuerdo al caso de uso diseñado, utilizando los componentes de ExtJS. El paquete de componentes Controlador contiene las acciones, las cuales responden a los servicios requeridos por los casos de uso a través del componente controlador frontal, que es la única puerta de entrada y salida a la aplicación. En este proceso el controlador utiliza el archivo componente Symfony. En la carpeta “lib” se encuentra el paquete de componentes de clases que asocian las acciones definidas por el caso de uso con la clase de la capa del

modelo que mapea la tabla correspondiente de la base de datos patdsi3. El paquete de componentes Modelo utiliza Propel para el trabajo con los datos, donde este facilita la labor de desarrollo de aplicaciones web.

3.2 Código Fuente

El código fuente son líneas de texto que nos permiten comprender y controlar las reglas implícitas en el funcionamiento de los programas, son las instrucciones que debe seguir la computadora para ejecutar dicho programa. Estas instrucciones son escritas en un lenguaje de programación, lenguaje que consiste en un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones (27).

3.2.1 Estándares de Codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vista a generar un código de alta calidad, es de gran importancia para la calidad del software y para obtener un buen rendimiento.

Estilo de Codificación Utilizado

- Todas las etiquetas php deben ser completas (<?php?>)... no reducidas (<? ?>).
- Los bloques de código siempre deben estar encerrados por llaves, si consta de una línea no es necesario utilizar llaves.
- Todas las funciones y clases deben estar comentadas. Los comentarios deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones o variables.
- Tamaño = 4 (espacios) para:
    - Declaraciones dentro de las clases.
    - Enunciado dentro de métodos y funciones.
    - Enunciados dentro de bloques de comandos.

Ejemplo de Código Fuente

A continuación se muestra un fragmento de código del método “calcularIndicadores” de la clase “TWPTypeModel” y se ofrece una breve descripción del mismo.

```
calcularIndicadores: function(paginas){
    //recorro las paginas del modelo
    for (posPag = 0; posPag < paginas.length; posPag++) {
        //recorro las indicadores de las paginas
        for (iii = 0; iii < this.tienenFormula(paginas,posPag); iii++)
            for (posInd = 0; posInd < paginas[posPag].indicadorespagina.length; posInd++) {
                //verifico si tiene formula

                if (paginas[posPag].indicadorespagina[posInd].formula!="no") {
                    //verifico si se puede calcular este indicador
                    var formula= paginas[posPag].indicadorespagina[posInd].formula.split(" ");
                    var indicadoresid=new Array();
                    for (i = 1; i < formula.length; i++) {
                        if (formula[i][0]=="[") {

                            indicadoresid.push(formula[i]);
                        }
                    }

                    var codigosdeFila= new Array();

                    for (i = 0; i < indicadoresid.length; i++) {
                        for (j = 0; j < paginas[posPag].indicadorespagina.length; j++) {
                            if ("["+paginas[posPag].indicadorespagina[j].idcodindicador+"]"
                                ==indicadoresid[i]) {
                                codigosdeFila.push(paginas[posPag].indicadorespagina[j].codfila);
```

Ilustración 10: Fragmento de código fuente del método "calcularIndicadores".

Este método es el encargado de calcular todos aquellos indicadores que tienen asociada una fórmula y de salvar los resultados en el DTO.

3.3 Pruebas

Las pruebas de software forman parte de lo que podemos denominar verificación y validación del software. La verificación se refiere al hecho de comprobar si estamos desarrollando el software correctamente. La validación se encarga de comprobar si estamos construyendo el producto correcto. Son un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación. Interesa señalar que en cada fase del ciclo de vida de desarrollo de software se plantea un conjunto de pruebas que permiten constatar que el software desarrollado satisface las especificaciones de esa fase (28).

Las pruebas son aplicadas para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo. Se distinguen los siguientes **niveles de pruebas:**

- Prueba de desarrollador

- Prueba independiente
- Prueba de Unidad
- Prueba de Integración
- Prueba de sistema
- Prueba de aceptación

En cada uno de los niveles antes expuestos se pueden ejecutar diferentes **tipos de pruebas:**

Funcionalidad

- **Función:** Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso.
- **Seguridad:** Asegurar que los datos o el sistema solamente es accedido por los actores deseados.
- **Volumen:** Enfocada en verificando las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la BD.

Usabilidad

Prueba enfocada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto y en línea, asistente documentación de usuarios y materiales de entrenamiento.

Fiabilidad

- **Integridad:** Enfocada a la valoración de la robustez (resistencia a fallos).
- **Estructura:** Enfocada a la valoración a la adherencia a su diseño y formación. Este tipo de prueba se realiza a las aplicaciones Web asegurando que todos los enlaces están conectados, el contenido deseado es mostrado y no hay contenido huérfano.
- **Stress:** Enfocada a evaluar cómo el sistema responde bajo condiciones anormales (extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponible).

Rendimiento

- **Benchmark:** es un tipo de prueba que compara el rendimiento de un elemento nuevo o desconocido a uno de carga de trabajo de referencia conocido.
- **Contención:** Enfocada a la validación de las habilidades del elemento a probar para manejar aceptablemente la demanda de múltiples actores sobre un mismo recurso (registro de recursos, memoria, etc.)

- **Carga:** Usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante. La variación en carga es simular la carga de trabajo promedio y con picos que ocurre dentro de tolerancias operacionales normales.
- **Performance profile:** Enfocadas a monitorear el tiempo en flujo de ejecución, acceso a datos, en llamada a funciones y sistema para identificar y direccional los cuellos de botellas y los procesos ineficientes.

Soportabilidad

- **Configuración:** Enfocada a asegurar que funciona en diferentes configuraciones de hardware y software. Esta prueba es implementada también como prueba de rendimiento del sistema.
- **Instalación:** Enfocada a asegurar la instalación en diferentes configuraciones de hardware y software bajo diferentes condiciones (insuficiente espacio en disco, etc.)

Para llevar a cabo las pruebas se utilizan **métodos de pruebas**, el método de caja negra y el método de caja blanca:

- La prueba de **caja negra** se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene.
- La prueba de la **caja blanca** del software comprueba los caminos lógicos del software proponiendo casos de prueba que ejerciten conjuntos específicos de condiciones y/o bucles. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado o mencionado.

Para comprobar el correcto funcionamiento del componente y detectar posibles fallas se realizarán pruebas de funcionalidad y usabilidad, utilizando el método de caja negra, a nivel de desarrollador y de integración quedando definida así la estrategia de prueba.

Prueba funcional:

Se aplica la prueba funcional, comprobando la interacción del usuario con la aplicación a medida que se prueba la correspondencia entre las funciones implementadas con los requisitos del cliente. Para ello se diseñaron casos de pruebas basados en los casos de uso y a su vez en los requisitos funcionales,

comparando cada funcionalidad implementada con la descrita, para verificar hasta qué punto cumplía con las necesidades del cliente. Se comprobó la correcta validación de los campos, verificando que cada cual aceptara exclusivamente los caracteres válidos. Por ejemplo en los campos donde su composición sólo era de números se verificó si el sistema permitía entrar letras o algún otro símbolo, dígase /, *, -, +, %, entre otros. De esta forma se aplicó simultáneamente la técnica de Partición de Equivalencia del Método de Caja Negra, con la comprobación de las variables válidas e inválidas. Cada defecto encontrado se fue registrando en la plantilla de no conformidades, argumentando cada no conformidad y clasificándola según su grado de importancia en significativa o no.

A continuación se muestra el diseño de casos de prueba del CU Gestionar Fórmulas.

Descripción de las variables:

No	Nombre del campo	Clasificación	Valor nulo	Descripción
1	Indicador	Campo de selección.	No	Campo de selección. Se selecciona en una tabla el indicador que se desea agregar a la fórmula.
2	Operador	Campo de selección.	No	Campo de selección. Se selecciona el operador que se desea agregar a la fórmula.
3	Constante	Campo de texto	No	Cadena de texto con longitud menor que 10, solo permite números, ya que es una constante que se le agrega a la fórmula.
4	Fórmula	Campo de texto	No	Cadena de texto, permite operadores, indicadores y constantes de forma tal que

				conformen una fórmula que pueda evaluarse.

Tabla 3: Descripción de las variables del diseño de casos de prueba del CU Gestionar Fórmulas.

Esta descripción posibilitó que se realizara una matriz de datos, donde se evaluó y probó la validez de cada uno de los datos introducidos en el sistema, específicamente en la sección que se estuvo probando. Utilizando un juego de datos válidos e inválidos se registraron, con el empleo de la técnica de partición de equivalencia, los resultados de las pruebas. La partición equivalente es un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico.

Descripción general

Este caso de uso tiene la finalidad de gestionar todo lo referente a las fórmulas de los indicadores.

Escenario	Descripción	1	2	3	4	Respuesta del sistema	Flujo central
EC 1.1 Asignar fórmula.	Se asigna una fórmula con datos correctos.	V	V	V	V	El sistema valida la fórmula y muestra mensaje de información.	El sistema guarda la fórmula.
EC 1.2 Asignar fórmula con datos incorrectos.	Se asigna una fórmula con datos incorrectos.	V	V	V	I	El sistema valida la fórmula y muestra mensaje de error.	El sistema desecha la fórmula.
EC 1.3 Cancelar operación	Cancela operación.	NA	NA	NA	NA	El sistema cancela la opción asignar fórmula.	El sistema no hace ningún cambio.

Tabla 4: Caso de prueba Asignar Fórmula del CU Gestionar Fórmula.

Escenario	Descripción	1	2	3	4	Respuesta del sistema	Flujo central
EC 2.1 Eliminar fórmula.	Se elimina la fórmula.	V	V	V	NA	El sistema muestra mensaje de información.	El sistema elimina la fórmula.
EC 2.2 Cancelar operación.	Cancela operación.	NA	NA	NA	NA	El sistema cancela la opción eliminar fórmula.	El sistema no hace ningún cambio.

Tabla 5: Caso de Prueba Eliminar Fórmula del CU Gestionar Fórmula.

Escenario	Descripción	1	2	3	4	Respuesta del sistema	Flujo central
EC 3.1 Modificar fórmula.	Se modifica la fórmula con datos correctos.	V	V	V	V	El sistema valida la fórmula y muestra mensaje de información	El sistema modifica la fórmula.
EC 3.2 Modificar fórmula con datos incorrectos.	Se modifica una fórmula con datos incorrectos.	V	V	V	I	El sistema valida la fórmula y muestra mensaje de error.	El sistema desecha la fórmula.
EC 3.3 Cancelar operación	Cancela operación.	NA	NA	NA	NA	El sistema cancela la opción modificar fórmula.	El sistema no hace ningún cambio.

Tabla 6: Caso de Prueba Modificar Fórmula del CU Gestionar Fórmula.

Escenario	Descripción	1	2	3	4	Respuesta del sistema	Flujo central
EC 4.1 Listar fórmula.	Se muestra la fórmula.	NA	NA	NA	NA	El sistema muestra la fórmula.	El sistema no hace ningún cambio.

Tabla 7: Caso de Prueba Listar Fórmula del CU Gestionar Fórmula.

Se obtuvo una no conformidad después de realizar estas pruebas. La misma consiste en que al asociar una fórmula a un determinado indicador pueda darse el caso de que nunca se resuelva, ejemplo:

A = C + B

B = A + 1

C = 1

En este caso el indicador A depende del B y del C, y el indicador B depende del A, lo que crea una dependencia recursiva entre estos dos indicadores impidiendo que estas fórmulas puedan ejecutarse.

Prueba de Integración:

A medida que se fue desarrollando el componente y que se implementaban cada una de sus funcionalidades se iba comprobando el efecto del mismo en SIGE, buscando posibles fallas y verificando su correcto funcionamiento.

Después de que se implementaba una funcionalidad se probaban todas las funcionalidades de los módulos en busca de fallas, estas pruebas detectaron varias no conformidades las cuales ya fueron resueltas.

Prueba de Usabilidad:

Para la realización de esta prueba se muestra al cliente el producto terminado para que interactúe con el mismo y de su opinión. El cliente determinó que el componente era fácil de usar y que no era necesaria la creación de un manual de usuario para trabajar con el mismo.

Resultados finales de las pruebas

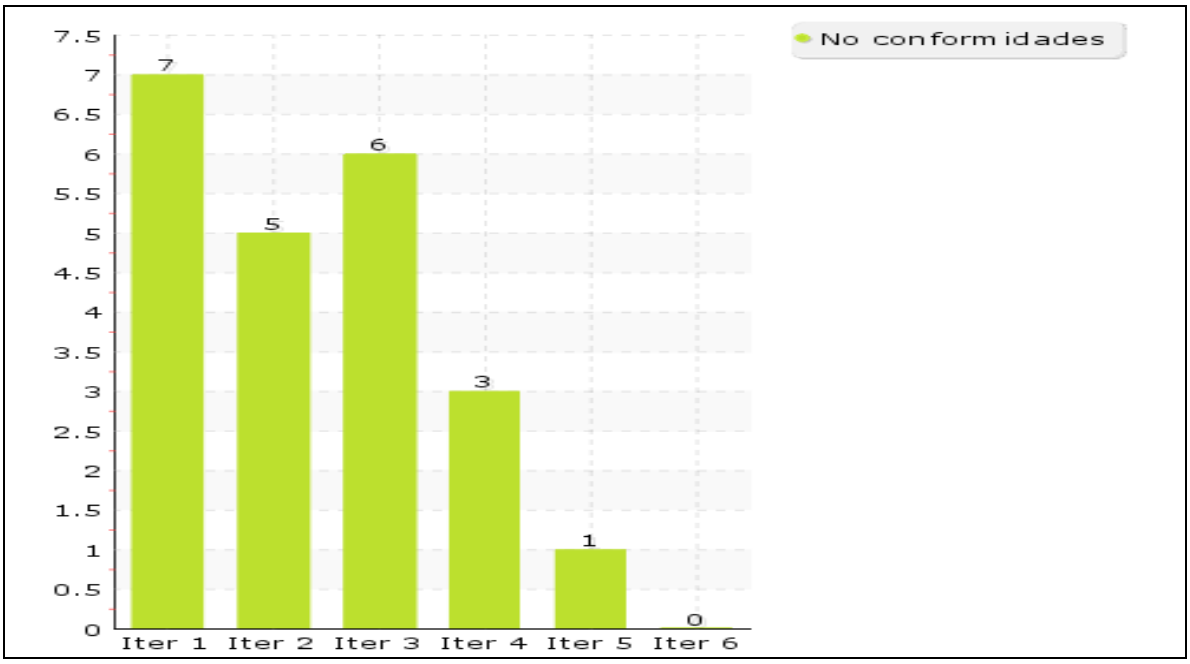


Ilustración 11: Resultados de las pruebas realizadas.

Las pruebas descritas en el documento corresponden a la iteración 5, donde solo se obtuvo una no conformidad.

3.4 Secciones principales de la interfaz gráfica

A continuación se muestran algunas secciones de la interfaz gráfica de la aplicación encaminadas a lograr una visión agradable, sencilla y atractiva mediante la utilización de las librerías ExtJS.

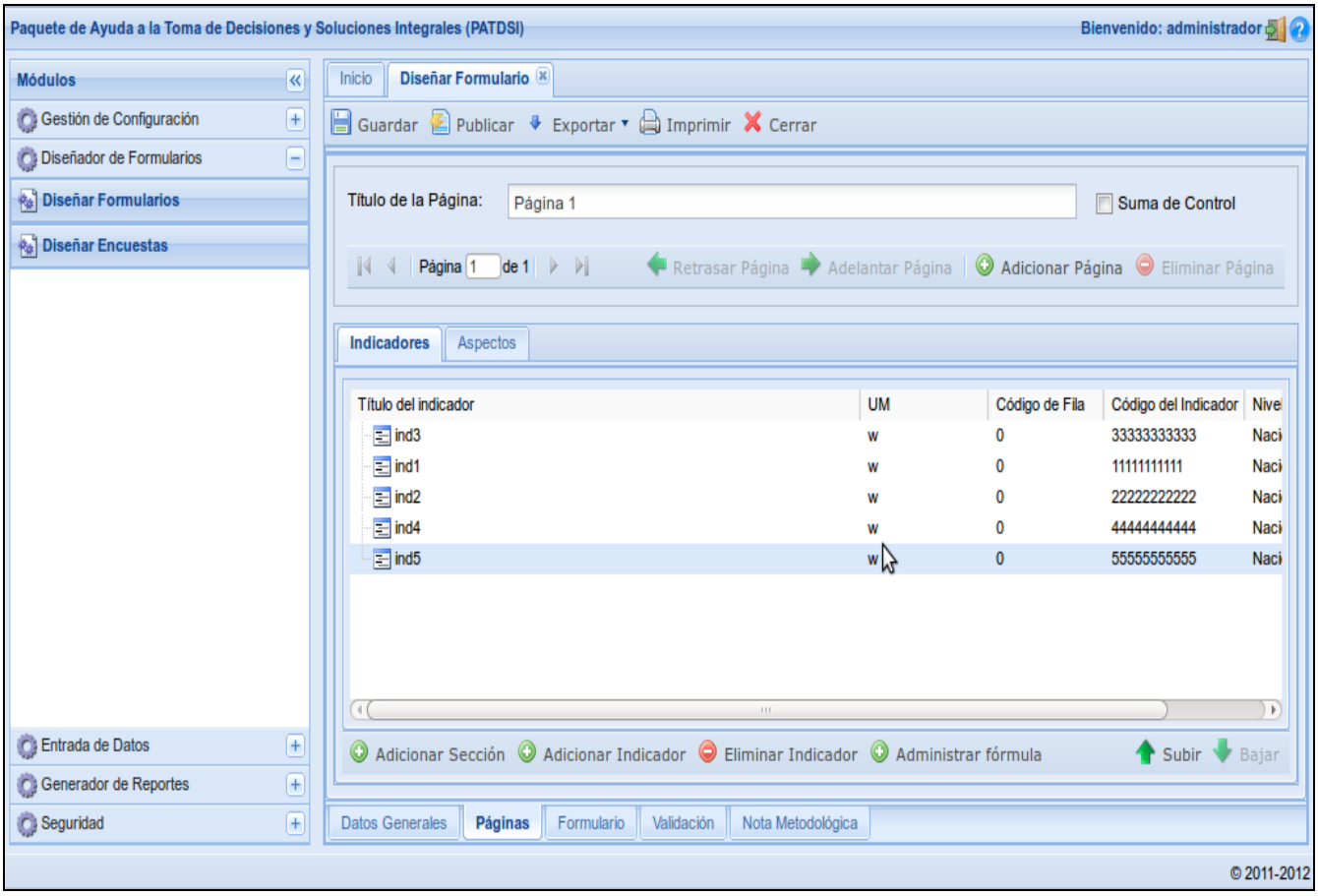


Ilustración 12: Interfaz "Diseñar Formularios".

La figura muestra la interfaz gráfica correspondiente a la funcionalidad diseñar formularios, la cual sufrió cambios para que el componente pudiera integrarse al sistema. Se hacen modificaciones que permitan al usuario final poder seleccionar el indicador al cual desean administrarle fórmula. Se modificaron los objetos de transferencia de datos de esta vista para que soportaran un nuevo campo (campo que almacena la fórmula) lo cual permite que la fórmula sea enviada satisfactoriamente a la clase controladora encargada de salvar la misma en la base de datos, se le agrega la opción "Administrar Fórmulas" la cual muestra la interfaz gráfica "Administrar Fórmulas", la cual se muestra a continuación.

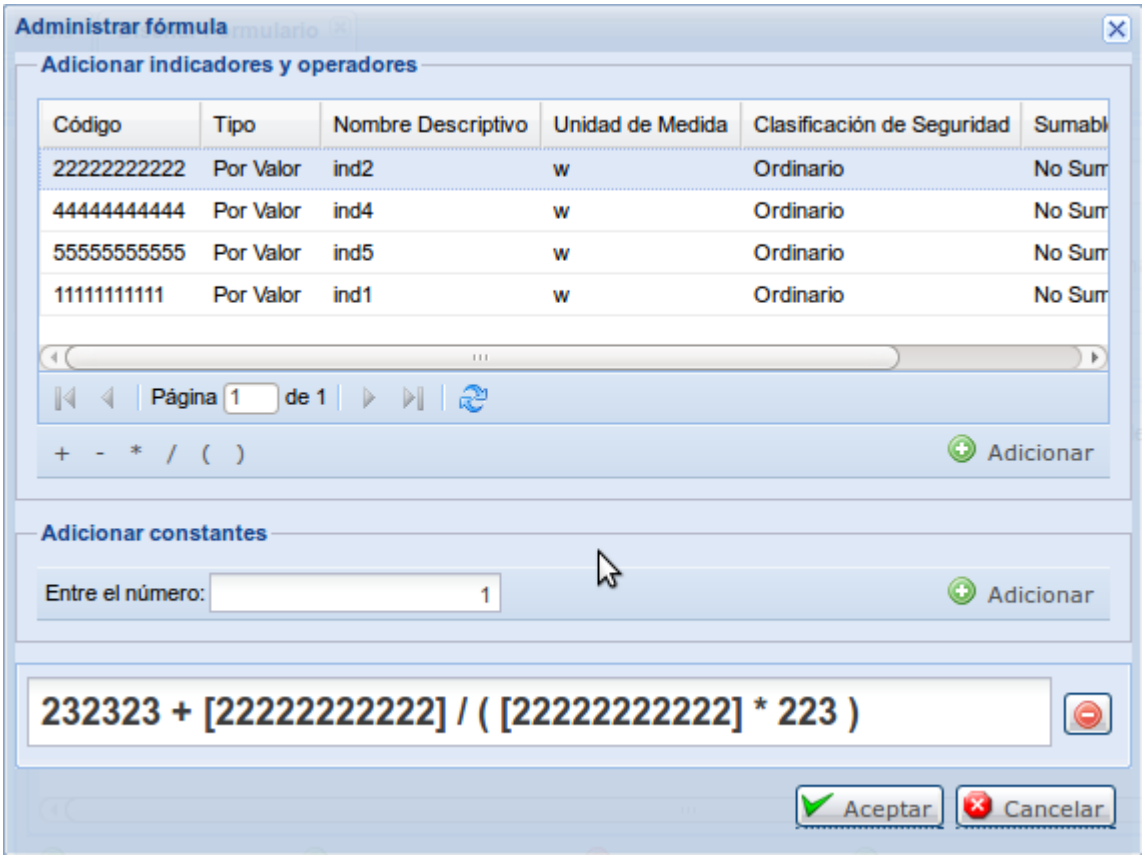


Ilustración 13: Interfaz "Administrar Fórmula".

La figura 11 muestra la interfaz gráfica correspondiente a la funcionalidad "administrar fórmula", la cual permite al usuario crear de manera fácil y eficiente la fórmula, limitando a casi cero los errores. Permite también modificar, listar y eliminar la fórmula de aquel indicador que tenga asignada una fórmula.

Conclusiones del Capítulo

La realización de este capítulo estructuró las clases del diseño en paquetes de componentes. Se definieron los estándares de codificación que permiten organizar el código a lo largo de la implementación. Se implementaron todas las funcionalidades garantizando su correcto funcionamiento mediante pruebas funcionales y de usabilidad utilizando el método de caja negra basado en la técnica de partición de equivalencia.

CONCLUSIONES GENERALES

Luego de realizada la investigación, se pudo arribar a las siguientes conclusiones:

- El estudio de los principales sistemas de gestión estadística garantizó una base metodológica sobre los elementos que deben caracterizar el proceso de gestión de fórmulas de indicadores.
- Al evaluarse los procesos de negocio asociados a la gestión de fórmulas de indicadores en SIGE se obtuvo un modelo guía para la implementación del sistema mediante la generación de los artefactos correspondientes a los flujos de trabajo propuestos por la metodología OpenUP.
- La implementación de las funcionalidades del componente, acorde a las pautas de diseño, garantizó el cumplimiento de lo establecido en la especificación de requisitos de software.
- La realización de pruebas funcionales y de usabilidad así como la resolución de las no conformidades encontradas demostraron el correcto funcionamiento del producto implementado.

RECOMENDACIONES

Después de haber alcanzado los objetivos que se trazaron al principio de este trabajo se propone la siguiente recomendación:

- Agregar al componente un editor de fórmulas, que permita editar cualquier elemento en la misma sin afectar a los demás elementos que la componen.

REFERENCIAS BIBLIOGRÁFICAS

1. **Desrosiéres, Alain.***La Política de los Grandes Números.* Barcelona : Melusina, 2004. ISBN: 84-933273-5-2.

2. **ONEI.** Oficina Nacional de Estadística e Información. [En línea] [Citado el: 3 de noviembre de 2012.] http://www.one.cu.

3. **Mayo, Marie Claude.***Informática Jurídica.* junio : Editorial Jurídica de Chile, 1991. ISBN:956-10-0916-0.

4. **Fernando Lamata Cotanda, Jose M Segovia de Arana.***Manual de administración y gestión sanitaria.* s.l. : Ediciones Díaz de Santos, 1998.

5. **Amaya, Jairo Amaya.***Gerencia: Planeación & Estrategia.* s.l. : Universidad Santo Tomas de Aquino, 2005.

6. **United Nations.***Encuestas de Hogares En Los Paises En Desarrollo y En Transicion.* s.l. : United Nations Publications, 2008. 9789213612255.

7. **Delgado, Cecilia Milián.***Sistema Integrado de Gestión Estadística (SIGE).* s.l. : Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, 2011.

8. **Romero, Dr. C. Raúl Rubén Fernández Aedo y Lic. Martín Enrique Delavaut.***EDUCACION Y TECNOLOGIA: Un binomio excepcional.* s.l. : Martín Delavaut. 9872323038, 9789872323035.

9. EcuRed. [En línea] [Citado el: 3 de diciembre de 2012.] http://www.ecured.cu/index.php/OpenUp.

10. EcuRed. [En línea] [Citado el: 3 de diciembre de 2012.] http://www.ecured.cu/index.php/UML.

11. **Tom Negrino, Dori Smith.***JavaScript and Ajax for the Web:Visual Quickstart Guide.* s.l. : Peachpit Press, 2008. 0321564081, 9780321564085.

12. **Fabien Potencier, François Zaninotto.***Symfony la guia definitiva.* 2008.

13. **Yera, Angel Cobo.***Diseño y programación de bases de datos.* Madrid : Visión Libros, 2007. ISBN:978-84-9821-459-8.

14. **PostgreSQL Cuba.** Acerca de PostgreSQL: PostgreSQL Cuba.*PostgreSQL Cuba.* [En línea] [Citado el: 7 de Diciembre de 212.] http://postgresql.uci.cu.

15. **VV Staff, VV.aa.***Windows Server 2003:preparacion para el examen MCSE, MSCA.* s.l. : Ediciones ENI, 2004. ISBN: 2-7460-2306-7.

16. **The Apache Software Foundation.** The Apache Software Foundation. [En línea] [Citado el: 8 de Diciembre de 2012.] http://www.apache.org/.

17. **Muñoz, Vicente Javier Eslava.***Aprendiendo a programar paso a paso con C.* s.l. : Bubok, 2012. 8468610623, 9788468610627.

18. **NetBeans IDE.** NetBeans IDE. [En línea] [Citado el: 8 de Diciembre de 2012.] http://netbeans.org.

19. **Weitzenfeld, Alfredo.***Ingeniería de software orientada a objetos con UML, Java e Internet.* s.l. : Cengage Learning Editores, 2005. 9706861904.

20. **Toni Granollers i Saltiveri, Jesús Lorés Vidal, José Juan Cañas Delgado.***Diseño de sistemas interactivos centrados en el usuario.* s.l. : Editorial UOC, 2005. 9788497882675.

21**.Sommerville, Ian.***Ingeniería del software. Fuera de colección.* s.l. : Pearson Educación, 2005. 8478290745,.

22. **Areba, esús Barranco de.***Metodología Del Análisis Estructurado de Sistemas.* s.l. : Univ Pontifica Comillas, 2002. 9788484680437.

23. **Sommerville, Ian.***Ingeniería del software 7/e.* s.l. : Pearson Educación, 2005. 8478290745.

24. **Andrés Gómez de Silva Garza, Ignacio de Jesús Ania Briseño.***Introducción a la Computación.* s.l. : Cengage Learning Editores, 2008. 9789706867681.

25. **Ernest Teniente López, Antoni Olivé Ramon, Enric Mayol Sarroca, Cristina Gómez Seone.***Diseño de sistemas software en UML.* Catalunya : Univ. Politè, 2004. 8498800757.

26. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.***El Proceso Unificado de Desarrollo de Software.* España : Adison-Wesley.

27. *SOFTWARE LIBRE.* s.l. : Icaria Editorial, 2008. 9788474269598.

28. **Fernando Alonso Amo, Loïc A. Martínez Normand, Francisco Javier Segovia Pérez.***Introducción a la ingeniería del software.* s.l. : Delta Publicaciones, 2005. 9788496477001.

29. **Sánchez, Linet Lores y Roque, Diana Monné.** *Aplicación de las pruebas de liberación al Sistema Informático de Genética Médica.* Ciudad de la Habana, Junio 2009. Trabajo de Diploma para optar por el título de Ingeniero Informático.

BIBLIOGRAFÍA

Aaron Gustafson, Jonathan Snook, Dan Webb, Stuart Langridge. *Accelerated DOM Scripting with Ajax, APIs, and Libraries.* s.l. : Apress, 2007. 143020284X, 9781430202844.

Allan Kent, Steven NOWICKI. *Fundamentos PHP 5.* s.l. : Anaya Multimedia, 2005. 844151805X, 9788441518056.

Amaya, Jairo Amaya. *Gerencia: Planeación & Estrategia.* s.l. : Universidad Santo Tomas de Aquino, 2005.

Andrés Gómez de Silva Garza, Ignacio de Jesús Ania Briseño. *Introducción a la Computación.* s.l. : Cengage Learning Editores, 2008. 9789706867681.

Areba, Jesús Barranco. *Metodología Del Análisis Estructurado de Sistemas.* s.l. : Univ Pontifica Comillas, 2002. 9788484680437.

Cobo, Ángel. *PHP y MySQL: Tecnología para el desarrollo de aplicaciones web.* s.l. : Ediciones Díaz de Santos, 2005. 8479787066, 9788479787066

Delgado, Cecilia Milián. *Sistema Integrado de Gestión Estadística (SIGE).* s.l. : Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas, 2011.

Desrosiéres, Alain. *La Política de los Grandes Números.* Barcelona : Melusina, 2004. ISBN: 84-933273-5-2.

Drozdek, Adam. *Estructuras de datos y algoritmos con Java.* s.l. : Cengage Learning Editores, 2007. 9706866116, 9789706866110.

Ernest Teniente López, Antoni Olivé Ramon, Enric Mayol Sarroca, Cristina Gómez Seone. *Diseño de sistemas software en UML.* Catalunya : Univ. Politè, 2004. 8498800757.

Fabien Potencier, François Zaninotto. *Symfony la guia definitiva.* 2008.

Fernando Alonso Amo, Loïc A. Martínez Normand, Francisco Javier Segovia Pérez. *Introducción a la ingeniería del software.* s.l. : Delta Publicaciones, 2005. 9788496477001.

Fernando Lamata Cotanda, Jose M Segovia de Arana. *Manual de administración y gestión sanitaria.* s.l. : Ediciones Díaz de Santos, 1998.

García, Isamel Gutiérrez. *Matemáticas para informática.* s.l. : Universidad del Norte. 9587410750, 9789587410754.

García, Jesús. *Ext JS in Action.Manning Pubs Co Series.* s.l. : Manning Publications Company, 2011. 1935182110, 9781935182115.

Geudens, Tom. *Resource-Oriented Computing with NetKernel: Taking REST Ideas to the Next Level.* s.l. : O'Reilly Media, Inc., 2012. 1449322484, 9781449322489.

Gómez, Carlos Barco. *Elementos de lógica.* s.l. : Universidad de Caldas. 958804197X, 9789588041971.

Gosselin, Don. *The Web Technologies Series.* s.l. : Cengage Learning, 2010. 0538748877, 9780538748872.

Herrington, Jack D. *PHP Hacks: Tips & Tools For Creating Dynamic Websites.*

Heurtel, Olivier. *PHP 5.3 DESARROLLAR UN SITIO WEB DINAMICO E INTERACTIVO.* s.l. : Ediciones ENI, 2011. 2746066661, 9782746066663.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. *El Proceso Unificado de Desarrollo de Software.* España : Adison-Wesley.

Johnsonbaugh, Richard. *Matemáticas discretas.* s.l. : Pearson Educación, 2005. 9702606373, 9789702606376.

Knuth, Donald Ervin. *El arte de programar ordenadores.* s.l. : Reverte, 1980. 8429126627, 9788429126624.

Leslie M. Orchard, Ara Pehlivanian, Scott Koon, Harley Jones. *Professional JavaScript Frameworks: Prototype,YUI, ExtJS, Dojo and MooTools.* s.l. : John Wiley & Sons, 2010. 0470555343, 9780470555347.

Lurig, Mario. *Php Reference: Beginner to Intermediate Php5.* s.l. : Mario Lurig, 2008. 143571590X, 9781435715905.

MANO, M. MORRIS. *Arquitectura de computadoras 3ED.* s.l. : Pearson Educación, 1994. 9688803618, 9789688803615.

Mayo, Marie Claude. *Informática Jurídica.* junio : Editorial Jurídica de Chile, 1991. ISBN:956-10-0916-0.

Maza, Miguel Ángel Sánchez. *JavaScript. Certificado de profesionalidad.* s.l. : Innovación Y Cualificación, 2012. 8495733188, 9788495733184

Muñoz, Vicente Javier Eslava. *Aprendiendo a programar paso a paso con C.* s.l. : Bubok, 2012. 8468610623, 9788468610627.

NetBeans IDE. NetBeans IDE. [En línea] [Citado el: 8 de Diciembre de 2012.] http://netbeans.org.

Nixon, Robin. *Learning PHP, MySQL, and JavaScript.* s.l. : O'Reilly Media, Inc., 2009. 1449379303, 9781449379308.

ONEI. Oficina Nacional de Estadística e Información. [En línea] [Citado el: 3 de noviembre de 2012.] http://www.one.cu.

Partrick Carey, Frank Canovatchel. *JavaScript. New perspectives series.* s.l. : Cengage Learning, 2006. 0619267976, 9780619267971.

Pedro Isasi Viñuela, Pedro Isasi Viñuela. *Lenguajes, gramáticas y autómatas: un enfoque práctico.* s.l. : Pearson Educación, 1997. 8478290141, 9788478290147.

Perry, Jason Michael. *ExtJS.* s.l. : Oreilly & Associates Incorporated., 2013. 1449361978, 9781449361976.

PostgreSQL Cuba. Acerca de PostgreSQL: PostgreSQL Cuba.*PostgreSQL Cuba.* [En línea] [Citado el: 7 de Diciembre de 212.] http://postgresql.uci.cu.

Roman, J. Sancho San. *Lógica matemática y computabilidad.* s.l. : Ediciones Díaz de Santos, 1990. 8487189539, 9788487189531.

Romero, Dr. C. Raúl Rubén Fernández Aedo y Lic. Martín Enrique Delavaut. *EDUCACION Y TECNOLOGIA: Un binomio excepcional.* s.l. : Martín Delavaut. 9872323038, 9789872323035.

Sánchez, Linet Lores y Roque, Diana Monné. *Aplicación de las pruebas de liberación al Sistema Informático de Genética Médica.* Ciudad de la Habana, Junio 2009. Trabajo de Diploma para optar por el título de Ingeniero Informático.

Sedgewick, R. *Algoritmos en C++.* s.l. : Ediciones Díaz de Santos, 1995. 0201625741, 9780201625745.

Sommerville, Ian.*Ingeniería del software 7/e.* s.l. : Pearson Educación, 2005. 8478290745.

Sommerville, Ian.*Ingeniería del software. Fuera de colección.* s.l. : Pearson Educación, 2005. 8478290745,.

The Apache Software Foundation. The Apache Software Foundation. [En línea] [Citado el: 8 de Diciembre de 2012.] http://www.apache.org/.

Tom Negrino, Dori Smith. *JavaScript and Ajax for the Web:Visual Quickstart Guide.* s.l. : Peachpit Press, 2008. 0321564081, 9780321564085.

Toni Granollers Saltiveri, Jesús Lorés Vidal, José Juan Cañas Delgado. *Diseño de sistemas interactivos centrados en el usuario.* s.l. : Editorial UOC, 2005. 9788497882675.

United Nations. *Encuestas de Hogares En Los Paises En Desarrollo y En Transicion.* s.l. : United Nations Publications, 2008. 9789213612255.

Vaughn, Daniel. *Ext GWT 2.0: Beginner's Guide : Take the User Experience of Your Website to a New Level with Ext GWT.* s.l. : Packt Publishing Ltd, 2010. 1849511853, 9781849511858.

VV Staff, VV.aa.*Windows Server 2003:preparacion para el examen MCSE, MSCA.* s.l. : Ediciones ENI, 2004. ISBN: 2-7460-2306-7.

Weitzenfeld, Alfredo.*Ingeniería de software orientada a objetos con UML, Java e Internet.* s.l. : Cengage Learning Editores, 2005. 9706861904.

Yera, Angel Cobo. *Diseño y programación de bases de datos.* Madrid : Visión Libros, 2007. ISBN:978-84-9821-459-8.

Zammetti, Frank. *Practical Ext JS Projects with Gears Apresspod Series Expert's voice in Web development Practical Projects.* s.l. : Apress, 2009. 1430219246, 9781430219248.

ANEXOS

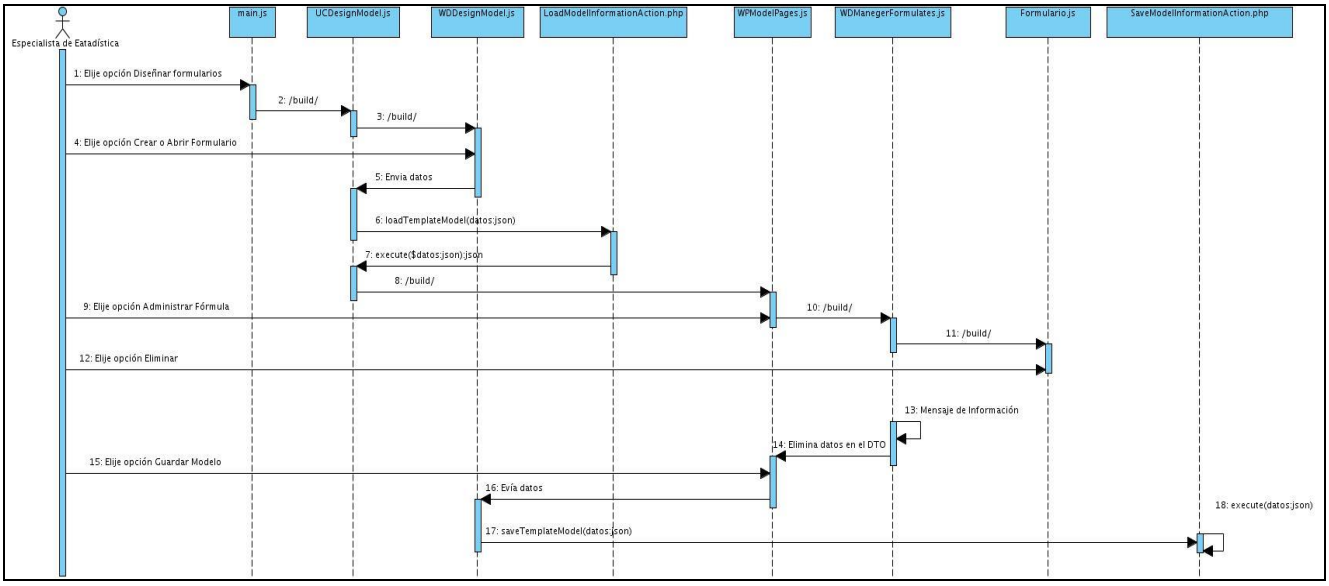


Ilustración 14: Diagrama de secuencia del escenario Eliminar Fórmula del CU Administrar Fórmula.

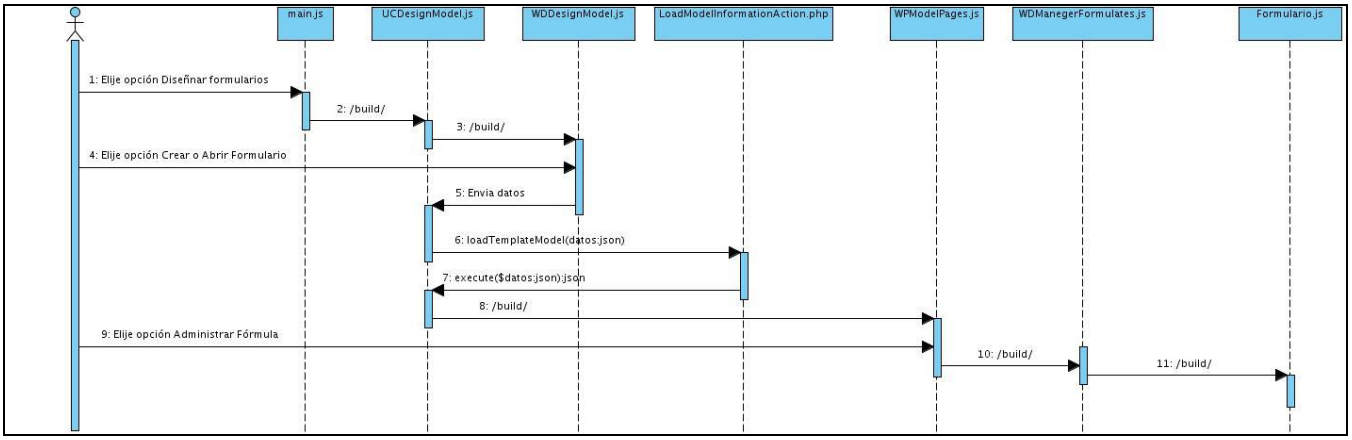


Ilustración 15: Diagrama de secuencia del escenario Listar Fórmula del CU Administrar Fórmula.

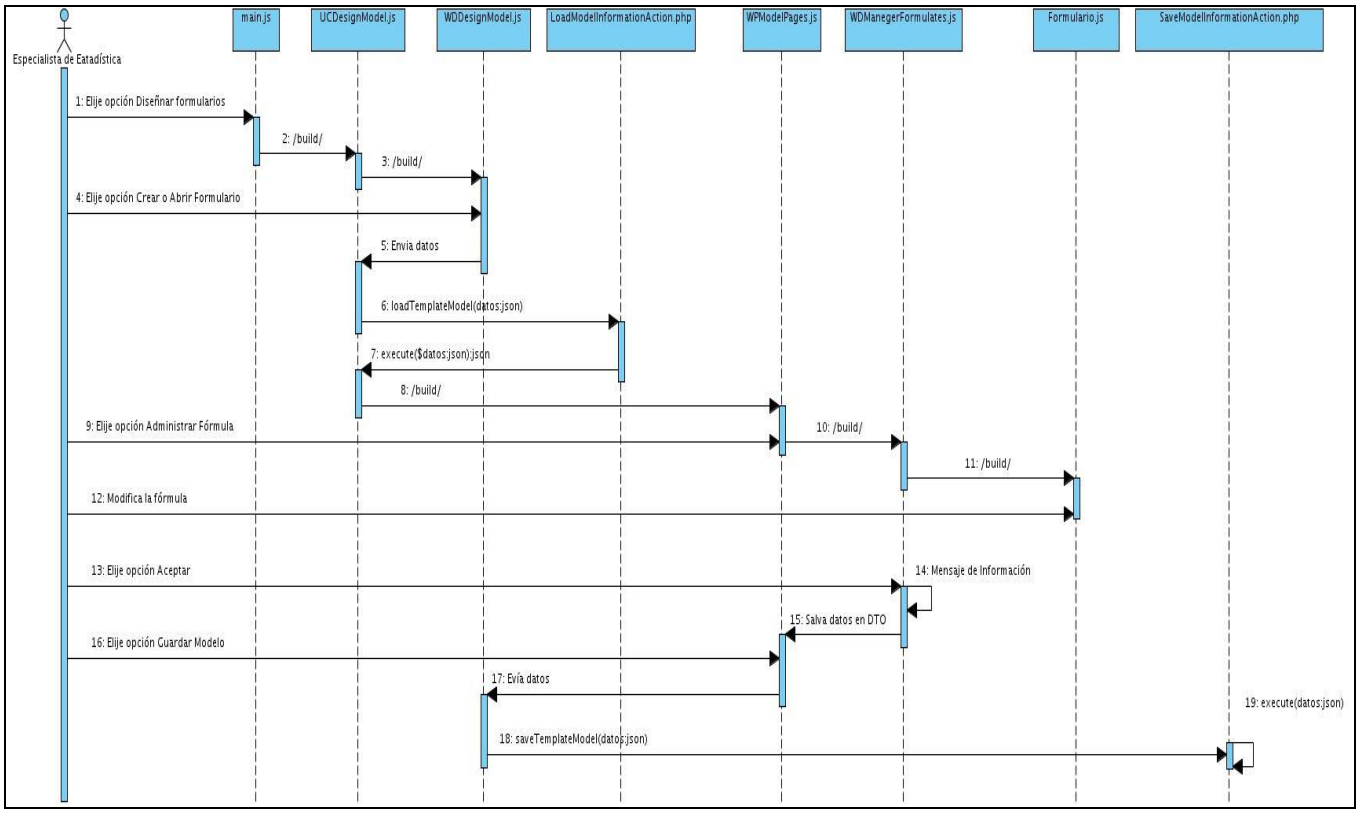


Ilustración 16: Diagrama de secuencia del escenario Modificar Fórmula del CU Administrar Fórmula.

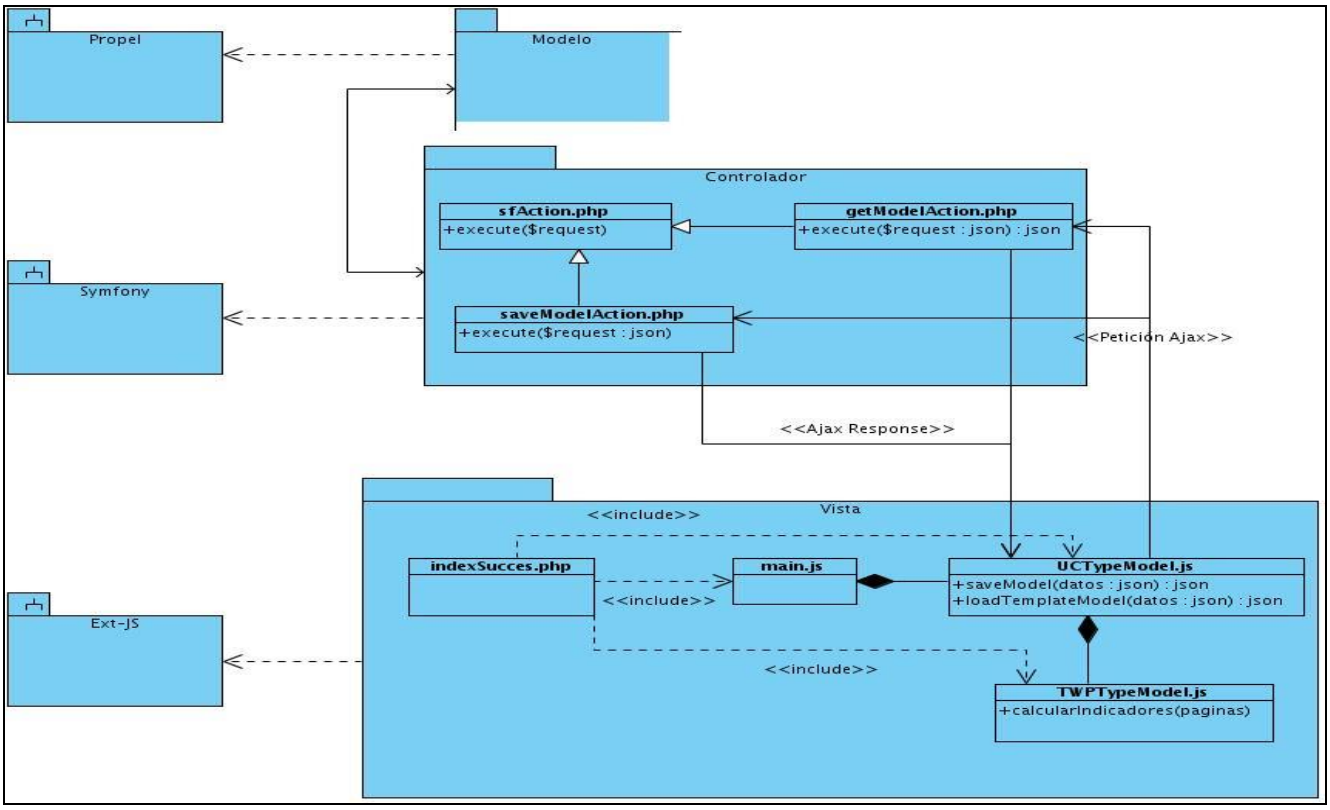


Ilustración 17: Diagrama de clases del diseño CU Calcular Indicadores.

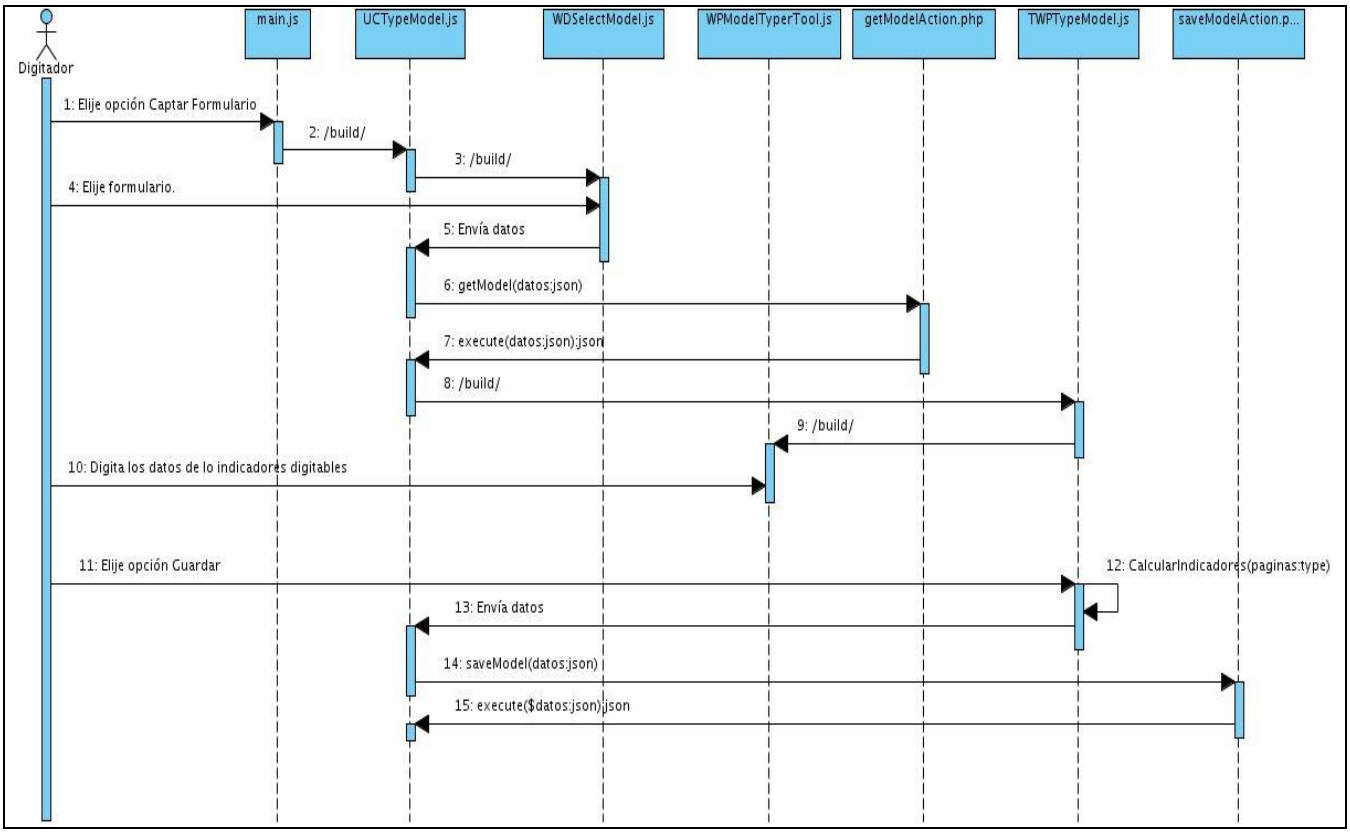


Ilustración 18: Diagrama de secuencia del CU Calcular Indicadores.

GLOSARIO DE TÉRMINOS

AJAX: (*Asynchronous Javascript And XML*), Javascript asíncrono y XML, es una técnica de desarrollo web para crear aplicaciones interactivas.

Apache: Es un software libre, servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

BD: Base de datos.

CASE: (Computer Aided Software Engineering). Ingeniería de Software Asistida por Computación.

CU: Caso de Uso.

DATEC: Centro de Tecnologías de Gestión de Datos.

DOM: (*Document Object Model*), Modelo de Objetos del Documento o Modelo en Objetos para la representación de Documentos, es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

GRASP: (*General Responsibility Assignment Software Patterns*), son patrones generales de software para asignación de responsabilidades. Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

Hardware: Componentes físicos que constituyen las computadoras y demás dispositivos periféricos.

HTML: (*Hyper Text Markup Language*). Lenguaje de Marcado de Hipertexto, es el lenguaje de marcado predominante para la elaboración de páginas web.

HTTP: (*Hypertext Transfer Protocol*).Protocolo de transferencia de hipertexto usado en cada transacción de la World Wide Web.

JQuery: Biblioteca o framework de Javascript, creada inicialmente que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

LINUX: Es un núcleo de sistema operativo libre tipo Unix. Es uno de los principales ejemplos de software libre.

MVC: Modelo Vista Controlador**,** es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

MySQL: Es un sistema de gestión de base de datos relacional.

ONE: Oficina Nacional de Estadísticas.

PATDSI: Paquete de Apoyo a la Toma de Decisiones en Sistemas Inteligentes.

PHP: (*Hypertext Preprocessor*), es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas.

PostgreSQL: Es un sistema de gestión de base de datos relacional orientada a objetos y libre.

Propel: Es un ORM para PHP que facilita la labor de desarrollo de aplicaciones web, gracias a la capa que transforma el tratamiento de la BD mediante objetos, con la que se puede recuperar, insertar y modificar datos.

TCP/IP: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP). El TCP/IP es la base de Internet, y sirve para enlazar computadoras que utilizan diferentes sistemas operativos.

UNIX: Es un sistema operativo de tiempo compartido, controla los recursos de una computadora y los asigna entre los usuarios. Permite a los usuarios correr sus programas. Controla los dispositivos de periféricos conectados a la máquina.

URL: (*Uniform Resource Locator***).** Localizador de Recurso Uniforme, dirección global de documentos y de otros recursos en la World Wide Web.

USB: Universal Serial Bus, puerto que sirve para conectar periféricos a una computadora.

WEB: World Wide Web también conocida como la web, el sistema de documentos o páginas web interconectados por enlaces de hipertexto, disponibles en Internet.

Windows: Es un sistema operativo con interfaz gráfica para computadoras personales propiedad de la empresa Microsoft. Windows es el sistema operativo más utilizado en el mundo.

XML: (*Extensible Markup Language*) Lenguaje de Marcas Extensible, es un metalenguaje extensible de etiquetas.