UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

SIGELITE DESKTOP: HERRAMIENTA DE CAPTURA DE DATOS PARA EL SISTEMA INTEGRADO DE GESTIÓN ESTADÍSTICA EN ENTIDADES CON LIMITACIONES TECNOLÓGICAS

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

Autores: Aylin Lugo Camacho

Alejandro García Hernández

Tutores: MSc. Alexis René Rodríguez León

Ing. Reynaldo Barceló Rodríguez

La Habana, Cuba, diciembre de 2014

"Año 56 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Aylin Lugo Camacho

_____

Firma del autor

Alejandro García Hernández

_____

Firma del autor

MSc. Alexis René Rodríguez León

_____

Firma del tutor

Ing. Reynaldo Barceló Rodríguez

_____

Firma del tutor

DATOS DE CONTACTO

Tutor

MsC. Alexis René Rodríguez León: Profesor graduado de Ingeniero en Ciencias Informáticas en el año 2007, en la Universidad de las Ciencias Informáticas. Se desempeña como profesor en la facultad 6 y como investigador en el Grupo de Bioinformática del Centro de Estudios de Matemática Computacional.

Contactar al correo arodriguezl@uci.cu

Tutor

Ing. Reynaldo Barceló Rodríguez: Profesor recién graduado de Ingeniero en Ciencias Informáticas en el año 2014, en la Universidad de las Ciencias Informáticas. Se desempeña como desarrollador del proyecto SIGE perteneciente al Centro de Tecnologías de Gestión de Datos (DATEC).

Contactar al correo barcelo@uci.cu

Resumen

En el Centro de Tecnologías de Gestión de Datos de la Universidad de las Ciencias Informáticas, se desarrolló el Sistema Integrado de Gestión Estadística (SIGE), el cual permite la Gestión de la Información en entidades u organismos. Para ello SIGE cuenta con diferentes módulos y en particular el Módulo de Entrada de Datos (MED) tiene como objetivo desarrollar el proceso de digitalización de los recursos (plantillas de formularios y de encuestas, así como formularios y encuestas digitalizadas), para cada unidad de observación es decir las entidades. Actualmente el proceso de recolección de datos estadísticos se realiza mediante SIGELITE, aplicación web que sustituyó el trabajo manual en las unidades de observación. A pesar de las mejoras que se lograron con este sistema, debido a las tecnologías con las que está desarrollado entra en conflicto con otras aplicaciones que utilizan las unidades de observación, provocando que el sistema se bloquee y con ello el retraso del trabajo. Además, los usuarios se quejan del sistema debido a errores existentes que no han sido solucionados por falta de soporte y porque han surgido nuevas necesidades en la captura de datos estadísticos que SIGELITE no satisface. El presente trabajo tiene como desarrollar una herramienta informática que sea desconectada, multiplataforma, que erradique los errores existentes en SIGELITE y permita mejorar el proceso de captura de datos en las unidades de observación. Se abordan además elementos esenciales como son: características, ventajas, arquitectura, herramientas y el análisis y diseño para lograr el desarrollo de la aplicación. Como resultado se obtuvo SIGELITE DESKTOP haciendo uso de las ventajas del entorno de escritorio que erradica las deficiencias del proceso de captura de datos estadísticos en las empresas del país.

Palabras claves: entorno de escritorio, Módulo de Entrada de Datos, SIGELITE.

Abstract

In the Data Management Technologies Center at the University of Informatics Sciences, it was developed an Statistics Management Integrated System (SIGE), which allows Information Management in entities or institutions. To do so, SIGE has different modules and in particular Data Entry Module (MED) that has as main objective to develop the process of digitization of resources (templates and survey forms and typed forms and surveys) for each Observing Unit that is to say entities. Currently the process of statistical data collection is done by SIGELITE, a web application which replaced manual work in the observation units. Despite the improvements achieved with SIGELITE, this system due to the technologies in which it was developed is not compatible with other applications that use the observation units, causing the system to crash and thus delaying the work. In addition, users complain about the system due to existing errors that have not been solved due to lack of support and because of the appearance, because for them being elders is difficult to adapt to technological changes. For these reasons, the objective of this work is to develop a software tool that is disconnected, multi-platform, with a friendly user interface to eradicate existing errors and allow improving data capture process in observation units. The present work also includes, essential elements such as: characteristics, advantages, architecture, tools and analysis and design to achieve the development of the application. As a result it was obtained a tool making use of the advantages of the desktop environment that eradicates the shortcomings of the statistical data capture process on the country's companies.

Keywords: Data Entry Module, desktop environment, SIGELITE.

Índice

Índice de Figuras

Índice de Tablas

Introducción

La sociedad impulsada por el avance científico tecnológico y apoyada en el uso generalizado de las Tecnologías de la Información y las Comunicaciones (TIC), ha experimentado cambios que alcanzan al mundo empresarial con el desarrollo de nuevos instrumentos, herramientas y métodos en apoyo a los procesos. Estas tecnologías se han convertido en medios indispensables de las instituciones, permitiendo ayudar, ahorrar trabajo y agilizar los procesos sustantivos. Es por esto que en Cuba, se aplican estrategias que tienen como objetivo desarrollar profundas transformaciones tecnológicas haciendo uso de las TIC, contando con el apoyo de instituciones como la Universidad de las Ciencias Informáticas (UCI) y el Centro de Tecnologías de Gestión de Datos (DATEC), perteneciente a esta institución.

DATEC juega un rol significativo en el desarrollo de sistemas de gestión de datos y tiene entre sus objetivos desarrollar bienes y servicios informáticos relacionados con esta temática. Estos sistemas brindan soporte a los procesos de toma de decisiones, planificación y control en muchas áreas importantes del país. Especialmente para la gestión estadística, DATEC ha desarrollado el Sistema Integrado de Gestión Estadística (SIGE), el cual tiene la visión de convertirse en una solución integral para el tratamiento de la información estadística en entidades y organismos del país.

SIGE es una aplicación web compuesta por 7 módulos, que tienen como finalidad la gestión de datos estadísticos de forma simple y ágil (SIGELITE: GESTIÓN ESTADÍSTICA EN ENTORNOS DESCONECTADOS O DE BAJAS PRESTACIONES, 2014). En particular el Módulo de Entrada de Datos (MED), es el encargado de la captura de información estadística a partir de plantillas y encuestas previamente definidas en el Módulo Gestor de Plantillas (MGP). Mediante el MED, se capturan datos estadísticos de los diferentes niveles organizacionales y de las entidades o empresas, las cuales se denominan unidades de observación. Actualmente SIGE se utiliza en organismos de la administración central del estado, tales como: el Tribunal Supremo Popular (TSP), la Fiscalía General de la República (FGR) y la Oficina Nacional de Estadísticas e Información (ONEI), considerada esta última su principal cliente, pues fue conjuntamente con esta entidad que la UCI desarrolló el sistema SIGE.

La ONEI es la encargada de organizar, proponer y ejecutar la estadística oficial en Cuba. Este organismo utiliza SIGE en los diferentes niveles (municipal, provincial y nacional), donde se disponen de los medios tecnológicos para el despliegue del sistema y del personal especializado para su uso. Por el contrario las

unidades de observación (entidades) en su generalidad, no tienen las condiciones necesarias para realizar la captura de información estadística mediante el MED, si no, que se ha creado un mecanismo donde cada unidad de observación, recopila los datos estadísticos empleando hojas de cálculo, procesadores de texto o documentos impresos. Esta información se entrega mensualmente y se procesa a nivel municipal, donde se digita empleando el MED de SIGE. El proceso de digitalización se realiza en un tiempo limitado y al concentrarse gran cantidad de datos y no contar con un formato uniforme, es muy común el retraso del trabajo y fallas al realizar el proceso, lo que repercute desfavorablemente en los resúmenes estadísticos a nivel del país y a la toma de decisiones.

Para dar solución a la situación anterior se desarrolló SIGELITE, una aplicación web conformada para el trabajo en escenarios de bajas prestaciones y sin comunicación con el servidor central. Este sistema tiene las funcionalidades del MED de SIGE en su versión 1.0, permitiendo que las unidades de observación, digiten sus datos y entreguen a la Oficina Municipal un archivo *XML*[1] el cual es compatible con SIGE, por lo que se ha logrado la distribución del trabajo estadístico, agilización del flujo informativo y ahorro de recursos.

Sin embargo, SIGELITE, a pesar de los beneficios que ofrece, no ha sido actualizado con las mejoras que brindan las recientes versiones del MED de SIGE, provocando que el mismo no satisfaga todas las necesidades que hoy demanda el proceso de captura de datos estadísticos en Cuba. Por otra parte, al ser una aplicación web utilizada de forma local y desconectada, para su despliegue y utilización debe utilizar otros programas y tecnologías como: Server2Go[2] (Timo Haberkern, 2015), Apache Portable Runtime[3] (The Apache Software Foundation, 2015), XAMPP[4] (Apache Friends, 2015) y Firefox[5] (Mozilla Foundation, 2015), que deben configurarse para utilizarla sobre Windows y las distribuciones de Linux. Esto provoca que se deban preparar varios paquetes y configuraciones en dependencia del sistema operativo que utilicen las unidades o empresas que deseen utilizar SIGELITE. También, en algunas unidades de observa-

[1] XML: eXtensible Markup Language (Lenguaje de Marcas Extensibles).
[2] Server2Go: es un servidor web que puede ejecutarse directamente desde un dispositivo de almacenamiento sin instalación.
[3] Apache Portable Runtime: es un mecanismo que permite ejecutar aplicaciones web en cualquier computadora sin necesidad de instalar algún programa
[4] XAMPP: es un servidor independiente de plataforma, actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas.
[5] Firefox: navegador web libre y de código abierto.

ción, SIGELITE entra en conflicto con otros programas que se utilizan para el trabajo estadístico, provocando que el sistema no se ejecute correctamente. Además, al no tener privilegios administrativos y/o conocimientos informáticos, los usuarios no pueden solucionar estos problemas, teniendo que recurrir a los especialistas informáticos, lo que trae como consecuencia el retraso del trabajo. De igual forma, muchos usuarios se quejan porque algunas de las funcionalidades centrales de SIGELITE demoran mucho tiempo en ejecutarse en unidades de observación con limitaciones tecnológicas y una vez dentro del sistema, el tiempo de respuesta de las operaciones que conllevan a la captura de datos se hace lento.

Por lo anteriormente planteado se identifica como **problema de la investigación**:

¿Cómo lograr una versión de SIGELITE que se adapte mejor a las necesidades que hoy demanda el proceso de captura de datos estadísticos en entidades con limitaciones tecnológicas?

Como **objeto de estudio**: el proceso de captura de datos en el Sistema Integrado de Gestión Estadística, y se define como **campo de acción:** el proceso de captura de datos en el Sistema Integrado de Gestión Estadística en entidades con limitaciones tecnológicas.

Para dar solución al problema descrito se define como **objetivo general de la investigación:** desarrollar una herramienta de escritorio que permita la captura de datos estadísticos para el Sistema Integrado de Gestión Estadística en entidades con limitaciones tecnológicas.

Del objetivo general formulado se derivan los **objetivos específicos:**

1. Caracterizar el proceso de captura de datos en SIGE.

2. Desarrollar una herramienta que permita la captura de datos estadísticos de manera compatible con SIGE.

3. Comprobar el funcionamiento de la solución propuesta.

Para darle cumplimiento al objetivo general de la investigación se definieron las siguientes **preguntas de investigación:**

- ¿Cuáles son los fundamentos teóricos del proceso de Captura de Datos en SIGE?

- ¿Cuáles son las características que debe cumplir la herramienta para la captura de datos estadísticos en entidades o empresas con limitaciones tecnológicas?
- ¿Cómo desarrollar una herramienta que permita la captura de datos estadísticos en entidades o empresas con limitaciones tecnológicas?
- ¿Cómo evaluar el correcto funcionamiento de la herramienta para la captura de datos estadísticos?

Con el fin de dar cumplimiento a los objetivos específicos se plantean las siguientes **tareas**:

1. Determinación de los elementos teóricos en los cuales se sustenta el proceso de captura de datos del Sistema Integral de Gestión Estadística (SIGE) para comprender mejor el funcionamiento de este módulo.
2. Análisis de las tecnologías, herramientas y metodología que utiliza SIGE para determinar las que intervienen en el proceso de desarrollo de la aplicación.
3. Estudio de las funcionalidades del MED de SIGE para la especificación de requisitos del sistema a implementar.
4. Análisis de los patrones arquitectónicos y de diseño para seleccionar los que se aplicarán a la solución.
5. Diseño del sistema para determinar los componentes que lo integran y la relación que existe entre los mismos.
6. Implementación de la aplicación para dar cumplimiento a los requisitos del sistema.
7. Realización de pruebas para comprobar el correcto funcionamiento del sistema.

Desde el punto de vista metodológico se emplearon los siguientes **métodos científicos**:

Métodos Teóricos:

- **Analítico – Sintético**: La utilización de este método permitió la descomposición del objeto de estudio en diferentes partes, las cuales fueron analizadas y de ahí surgió un nuevo conocimiento concreto, logrando desarrollar el fundamento teórico que sustenta la investigación.

- **Modelación:** Se utilizó en la elaboración de los diagramas correspondientes a los modelos de análisis, diseño e implementación de la herramienta a desarrollar.

Método Empírico:

- **Entrevista**: Se realizaron entrevistas a los especialistas del Sistema Integrado de Gestión Estadística y de la ONEI para obtener de forma concreta las necesidades que existen y poder desarrollar una herramienta con todas las funcionalidades que se necesitan.

El presente trabajo está estructurado por:

Resumen, Introducción, 3 Capítulos, Conclusiones, Recomendaciones, Glosario de términos, Bibliografía y Anexos. Los capítulos de la presente investigación se fundamentan en:

Capítulo 1: Fundamentación teórica: Este capítulo constituye el marco teórico de la investigación, se presentan un conjunto de conceptos y la síntesis de los resultados obtenidos en la revisión bibliográfica de acuerdo al tema de investigación. Se abordan los contenidos relacionados con la captura de datos. Se definen las herramientas, metodología de desarrollo, lenguajes de programación y las técnicas a utilizar para la creación de la solución.

Capítulo 2: Análisis y diseño de la aplicación: En este capítulo se describen las características de la aplicación a desarrollar. Se definen los requisitos funcionales y no funcionales, los casos de uso del sistema, los patrones arquitectónicos y de diseño empleados en el desarrollo de la aplicación. Se presentan los diagramas de clases con el objetivo de mostrar cada una de las funcionalidades que forman parte de la solución de forma detallada.

Capítulo 3: Implementación y pruebas de la aplicación: En este capítulo se presentan las principales tareas de implementación a realizar, se elaboran los artefactos correspondientes a la implementación. También se diseñan y aplican las pruebas para comprobar el funcionamiento de la aplicación.

Capítulo 1: Fundamentación teórica

En este capítulo se tratarán los aspectos teóricos de la investigación, abordando los conceptos asociados al dominio del problema para así lograr una mejor comprensión de la investigación. También son caracterizadas las técnicas, metodología, herramientas y tecnologías a utilizar con el fin de fundamentar su uso en la solución que se propone.

1.1 Conceptos básicos asociados a la investigación

Con el fin de que el lector pueda lograr una mejor comprensión de los temas que serán tratados en este capítulo, concisamente relacionados con el objeto de estudio de la investigación, se describen a continuación un grupo de conceptos vinculados al dominio del problema.

Encuesta

Las encuestas contienen un conjunto de preguntas elaboradas por especialistas y una vez aplicadas a los encuestados arrojan resultados que sirven para estudios posteriores. Una encuesta está conformada por las bases metodológicas, por el encabezado, que constituye un tipo especial de sección, por secciones las cuales contienen preguntas y estas a su vez contienen las variables. La encuesta tiene un ciclo de vida, inicialmente en el estado no activado se aplica el proceso de diseño de la encuesta, luego de aprobado el diseño, esta pasa a estado activado donde se pone en marcha al proceso de pre visualización de la encuesta, se publica y se le da un seguimiento mientras se encuentra activada la misma.

Formularios

En SIGE un formulario es un documento digital, diseñado con el propósito de que el usuario introduzca datos estructurados en las zonas del documento destinadas a ese propósito, para ser almacenados y procesados posteriormente. Un formulario está conformado por: el encabezado del formulario, cuerpo del formulario y herramienta del captador. El formulario tiene un ciclo de vida, inicialmente es enviado a digitación, luego a validación y finalmente a archivo. (Ver Anexo #1 Ejemplo de formulario en SIGE).

Unidades de Observación

La unidad de observación se considera como el elemento unitario del cual se obtienen datos con propósitos estadísticos sobre el conjunto al que pertenece. Cabe señalar que el tipo de unidad de observación está estrechamente ligado a la naturaleza del proyecto estadístico, pudiendo ser lugares, individuos, instituciones o eventos.

En SIGE se denomina unidades de observación a todas aquellas unidades presupuestadas, cooperativas, empresas mixtas y otras entidades, que son gestionadas en el Módulo Gestión de Configuración. Constituyen el centro de la gestión de configuración debido a que la información a captar corresponde a estas.

Indicadores

Los indicadores son herramientas para clarificar y definir de forma más precisa objetivos e impactos. Son medidas verificables de cambio o resultado, diseñadas para contar con un estándar contra el cual evaluar, estimar o demostrar el progreso con respecto a metas establecidas.

En SIGE los Indicadores son herramientas para clasificar y definir, de forma más precisa, objetivos e impactos. Forman parte de las Plantillas de Formulario a nivel de fila y se asocian a las temáticas las cuales permiten agruparlos.

Aspectos

Los aspectos son un conjunto de rasgos o características que muestra una persona o cosa. En SIGE los aspectos son elementos relacionados a los indicadores que enmarcan la dimensión de estos, entiéndase características más específicas. Forman parte de las Plantillas de Formulario a nivel de Columna**.**

1.2 Sistema Integrado de Gestión Estadística

El sistema SIGE fue desarrollado de conjunto con la Oficina Nacional Estadística e Información (ONEI). SIGE ha suplantando al sistema Microset NT existente en la ONEI que se comporta inestable fuera de MS-DOS y por su antigüedad no puede responder a las más revolucionarias tendencias del trabajo estadístico.

SIGE es una aplicación web enriquecida (RIA, según siglas en inglés), que tiene como finalidad la gestión de los datos estadísticos de forma simple y ágil. Su arquitectura web hace posible la recepción de la información desde cualquier parte y su consulta puede hacerse sin necesidad de estar físicamente en su origen. SIGE permite el diseño de formularios y encuestas para la captura de la información estadística asociada a las empresas; lo que contribuye al proceso de toma de decisiones al poderse evaluar el progreso de una empresa respecto a las metas establecidas. (Claudia Nuñez Sanz 1*, 2013)

Este sistema está compuesto por los siguientes módulos: Gestión de Configuración, Gestor de Plantillas, Entrada de Datos, Generador de Reportes, Herramientas, Administración y Seguridad. A continuación se describen algunos de ellos:

Gestión de Configuración (MGC)

Desarrolla el proceso de captación y procesamiento de la información estadística asociado a las unidades de observación, para lo cual dispone de registros que identifican a las distintas unidades de observación y clasificadores que permiten agruparlas y caracterizarlas en un lenguaje adecuado.

Gestor de Plantillas (MGP)

Es el encargado de la automatización del proceso de diseño de los formularios y encuestas necesarios para la captación de la información estadística en las unidades de observación. Constituye la entrada para el procesamiento y obtención de informes y reportes de comportamiento estadísticos para la toma de decisiones estratégicas en las diferentes esferas económicas del país. Permite además la visualización de las plantillas de encuestas y de formularios publicados, así como la modificación de las mismas si el elemento a modificar no afecta la estructura definida.

Entrada de Datos (MED)

El Módulo de Entrada de Datos ha surgido con el fin de encontrar una solución informática a las técnicas actuales de captación de datos. Este módulo es el encargado de la captura de la información estadística a partir de las plantillas de formularios y encuestas diseñadas en el Módulo Gestor de Plantillas y de las unidades de observación gestionadas en el Módulo Gestión de Configuración. Para cumplir con su objetivo, presenta cinco funcionalidades:

- Actualizar Formularios: esta funcionalidad permite la gestión de las plantillas de formularios y su habilitación o inhabilitación para la captación de la información. Los formularios captados, entiéndase información asociada a una unidad de observación en una plantilla de formulario, son gestionados de acuerdo a sus estados, que pueden ser: digitación (proceso de captación de la información), validación (proceso de evaluación del contenido del formulario de acuerdo a las reglas definidas en su diseño) y archivo (una vez que el formulario ha sido captado y validado). En cualquiera de estos estados, los formularios pueden ser eliminados del sistema, así como también exportados e importados hacia o desde archivos XML o MSET (formato similar al CSV, pero separado por espacios y con especificación a partir de la estructura de los formularios diseñados en el Módulo Diseñador de Formularios).

- Actualizar Encuestas: esta funcionalidad permite la gestión de las plantillas de encuestas y su habilitación o inhabilitación para la captación de la información. Las encuestas captadas, entiéndase información asociada a una unidad de observación en una plantilla de encuesta, son gestionadas de acuerdo a sus estados, que pueden ser: digitación (proceso de captación de la información), y archivo (una vez terminada la encuesta). En cualquiera de estos estados, las encuestas pueden eliminarse del sistema, así como también exportarse e importarse hacia o desde archivos XML.

- Captar Formularios: esta funcionalidad permite cargar una plantilla de formulario para la captura de datos a partir del mismo. Para ello, presenta una sección de Captación Rápida que dinámicamente se adecúa al formulario en cuestión guiando el proceso a través de los diferentes aspectos (columnas del formulario), con lo que se agiliza y humaniza el trabajo del usuario final. Contiene también otras secciones como la Vista Formulario que muestra en formato HTML una vista del formulario que se está captando con los datos que contiene hasta ese momento y la Nota Metodológica, que representa igualmente en formato HTML, la metodología asociada a la naturaleza y forma de captación de la plantilla.

- Captar Encuestas: esta funcionalidad es similar a la de Captar Formularios, sólo que sobre otro tipo de plantilla. Dado que las preguntas incluidas en una encuesta pueden ser sostenidas por diferentes formas de captación: campo de texto, campo numérico, campo de fecha, selección simple, selección múltiple y caja de texto, las mismas se cargan en una vista de impresión donde es capturada la respuesta de cada pregunta de acuerdo al caso correspondiente. También en este caso, existe una

sección Nota Metodológica análoga a la de Captar Formularios.

- Validar Formularios: como se había mencionado anteriormente, cada plantilla de formulario define reglas para validar la información. Esta funcionalidad verifica el cumplimiento de estas reglas. Una vez que ha sido realizada dicha verificación, el formulario puede transitar por 2 estados: archivo, en caso de que la validación haya resultado satisfactoria; o digitación, de ocurrir errores durante la verificación de cada regla definida, por lo que el formulario deberá cargarse en Captar Formularios para corregir los errores detectados.

Actualmente el sistema SIGE es utilizado por organismos de la administración central del estado como: la Fiscalía General de la República (FGR) y La Oficina Nacional de Estadísticas e Información (ONEI), siendo este último su principal cliente. La ONEI es la encargada de proponer, organizar y ejecutar la estadística oficial en Cuba. Este organismo a partir de la implantación del sistema SIGE realizada a nivel de país, logró sustituir un obsoleto sistema y con ello la necesaria modernización y mejora del flujo informativo en la ONEI. Sin embargo, el trabajo realizado por los digitalizadores en las Oficinas Municipales de Estadística e Información (OMEI) continuaba siendo complejo pues debían ser capaces de digitalizar en SIGE, en un tiempo limitado, toda la documentación entregada por las unidades de observación con sus respectivas estadísticas. Una propuesta para enfrentar esta problemática fue la de hacer que las propias unidades de observación fueran las encargadas de la digitación de sus estadísticas. Esto implicaba el acceso desde una empresa u organismo al SIGE instalado en la OMEI correspondiente o en el caso de no existir conectividad, la instalación del sistema. En cualquiera de los dos casos se requerirían de precondiciones tecnológicas imposibles de lograr o exigir en cada municipio del país.

Para dar solución a la situación anterior se desarrolló SIGELITE, una aplicación web conformada para el trabajo en escenarios de bajas prestaciones y sin comunicación con el servidor central.

SIGELITE

SIGELITE es una aplicación web, que tiene como objetivo sintetizar funcionalidades del Módulo de Captura de Datos de SIGE. Por las características de ser un sistema ligero (funcional en entornos de bajas prestaciones), desconectado (funcional con conectividad nula o deficiente), portable (funcional sin necesidad de instalarse) y multiplataforma, permite la captación de información de forma digital y directa de las uni-

dades de observación, así como que dichas unidades digiten y validen la información de todos los modelos correspondientes al Sistema de Información Estadística Nacional (SIEN). (Ing. Héctor Luis Reyes Zaldívar 1*, 2014)

SIGELITE se encuentra desplegado y en explotación en la ONEI y todas sus entidades a nivel nacional. El sistema presenta dos procesos fundamentales:

- Gestionar Recurso (los recursos son los formularios del sistema), el cual incluye las siguientes funcionalidades: Buscar recurso, Exportar recurso, Importar recurso, Eliminar recurso y Listar recurso.

- Captar Recurso, el cual incluye las siguientes funcionalidades: Captar recurso, Validar recurso, Visualizar formato del recurso, Visualizar Nota Metodológica del recurso y Guardar recurso.

Es válido señalar que la aplicación es compatible con los sistemas contables más empleados en el país, como son: VERSAT (BWT SIME, 2001), SISCONT5 (Centro de Investigaciones para la Industria Minero-Metalúgica, 2005), SISCOMIP (Juventud Rebelde, 1999), RODAS (CITMATEL, 2002), entre otros. Pero al requerir, para su despliegue y utilización, de varios paquetes y configuraciones en dependencia del sistema operativo, además de otras herramientas como: Server2Go, Apache Portable Runtime, XAMPP y Firefox, en varias unidades de observación SIGELITE entra en conflicto con algunos de estos sistemas, provocando el retraso del trabajo por la falta de privilegios administrativos y/o conocimientos informáticos de los especialistas. Además, SIGELITE no ha incorporado las nuevas capacidades que va demandando el proceso de captura de datos estadísticos en Cuba, como por ejemplo, la gestión de encuestas y las configuraciones generales de captación de formularios y encuestas.

Por otra parte, algunas de las funcionalidades centrales de SIGELITE, demoran mucho tiempo en ejecutarse en unidades de observación con limitaciones tecnológicas y en otros casos, la aplicación presenta errores en ciertas funcionalidades, como son:

- Importar y exportar plantillas: Al desarrollarse nuevas versiones de SIGE y no darle soporte a SIGELITE, este último no permite importar y exportar plantillas de las últimas versiones de SIGE.

- Limpiar fila: Durante el proceso de captación, en ocasiones, es necesario borrar todos los valores de los aspectos de una fila. En SIGELITE, no solo se borran estos valores sino que además se elimina la fila, lo que trae como consecuencia que no se pueda captar más datos sobre la misma.

- Configurar captación: solo se permite que un aspecto se relacione con otro y no con varios.

- Formato: las nuevas versiones de SIGE incluyen la posibilidad de exportar el formato del formulario a ficheros de tipo PDF[6] y SIGELITE no. Además, en algunos casos, SIGELITE no permite mostrar la imagen del formato de los formularios.

Propuesta de Solución

Se propone desarrollar una versión en entorno de escritorio de SIGELITE (en lo adelante SIGELITE DESKTOP), que sea multiplataforma y portable, incorporando nuevas funcionalidades para la gestión de encuestas, así como para las configuraciones generales de captación de formularios y encuestas. Además, se modificarán algunas funcionalidades que actualmente no funcionan correctamente en SIGELITE. El sistema deberá ajustarse al Manual de Normas Gráficas para los productos de la rama Sistema de Gestión Empresa-Industria: XEDRO (GESPRO. Paquete para la gestión de proyectos, 2013).

1.3 Ambiente de desarrollo

En el proceso desarrollo de software es fundamental la correcta elección de las tecnologías, herramientas y métodos que mejor se adapten y mayores facilidades brinden para dar solución al problema de la investigación. La selección de este ambiente de desarrollo se ajusta a la política de lograr un producto con la mayor calidad posible y a un costo reducido.

1.3.1 Metodología de desarrollo

Las metodologías de desarrollo de software surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un producto de software. No existe una metodología de desarrollo de software universal, pues las características de cada

[6] PDF: *Portable Document Format*.

proyecto (tiempo de duración, equipo de desarrollo, recursos) exigen que se escoja la metodología más adecuada para favorecer el trabajo de las personas que intervienen durante todo el proceso de desarrollo de software.

Una metodología es un conjunto integrado de técnicas y métodos que permite abordar de forma homogénea y abierta cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Es un proceso de software detallado y completo. Las metodologías se dividen en dos grupos, tradicionales o robustas y ágiles o ligeras. A continuación una breve comparación entre las metodologías ágiles y tradicionales (Hernández Sampieri, 1998):

Tabla 1. Comparación entre las metodologías ágiles y las metodologías tradicionales

Metodologías ágiles	**Metodologías tradicionales**
Especialmente preparados para cambios durante el proyecto. Proceso menos controlado, con pocos principios.	Cierta resistencia a los cambios Proceso mucho más controlado, con numerosas políticas/normas.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (menos de 10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Más artefactos.
Pocos roles.	Más roles.
XP, Open UP, SCRUM, Crystal, Feature -Driven Development (FDD), Adaptive Software	RUP, Métrica V3.

Development, DSDM.	

A continuación se describe la metodología de desarrollo que más se ajusta a las necesidades del equipo y al desarrollo de la aplicación, de manera que el proceso de desarrollo del software cumpla con la calidad requerida y se termine en el tiempo previsto para satisfacer al cliente con una aplicación que cumpla todas sus expectativas.

Metodología Proceso Unificado Abierto (OpenUP):

OpenUP es un proceso de desarrollo iterativo del software que es mínimo, completo, y extensible. El proceso es mínimo pues solamente la documentación fundamental es incluida; es completo ya que puede manifestarse como todo el proceso para construir un sistema; extensible dado que puede utilizarse como base sobre la cual el contenido del proceso se pueda agregar o adaptar según lo necesitado. (Canós, 2008)

Se decide usar esta metodología de desarrollo para la solución, debido a que al trabajar con los objetivos de días, semanas y meses para la programación de entregables, se logrará obtener una aplicación en el tiempo debido. Como el equipo de trabajo es pequeño, cada integrante tendrá que pasar por más de un rol y el sistema se podrá adaptar a los cambios que puedan aparecer en el proceso de desarrollo del software dado que está en continuo perfeccionamiento. Además de que permite la programación de entregables en todo el proceso iterativo con el cual se podrá arribar a mejores resultados con una buena comunicación con el cliente, ya que el mismo podrá ir comprobando si el software cumple los requisitos definidos en cada uno de los entregables de manera iterativa. Por otra parte,

Beneficios del uso de OpenUP (Eclipse, 2009)**:**

- 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.
- Se centra en una arquitectura temprana para reducir al mínimo los riesgos y organizar el desarrollo.
- Permite detectar errores tempranos a través de un ciclo iterativo.
- Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP.

- Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas.

1.3.2 Lenguaje de programación

Los lenguajes de programación son la vía de comunicación entre el hombre y la computadora. Un lenguaje de programación es una técnica estándar de comunicación. Permite expresar las instrucciones que han de ejecutarse en una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen un programa informático. A continuación, se describen las principales características del lenguaje de programación elegido para el desarrollo de la aplicación (Lenguajes de Programación, 2010).

Java

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo, lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es uno de los lenguajes de programación más populares en uso.

A continuación característica de Java (Java, 2011):

1. Lenguaje Orientado a Objetos.
2. Disponibilidad de un amplio conjunto de bibliotecas.
3. Lenguaje simple.
4. Distribuido.
5. Interpretado y compilado a la vez.
6. Robusto.
7. Seguro.

8. Indiferente a la arquitectura.
9. Portable.
10. Alto rendimiento.
11. Multihilo.
12. Dinámico.

Fundamentación de la selección

Se decide utilizar Java como lenguaje de programación debido a que es un lenguaje de programación libre, con la ventaja de poder usarlo en cualquier plataforma que cuente con la máquina virtual de Java instalada. Cuenta con una amplia documentación disponible para la comunidad de manera gratuita y una gran variedad de Interfaz de Programación de Aplicaciones (API por sus siglas en inglés) que facilitan el trabajo de los desarrolladores. Se seleccionó para la realización de SIGELITE DESKTOP además porque el lenguaje al ser multiplataforma permite el despliegue de la aplicación en cualquier plataforma, lo que hace posible una mayor flexibilidad a los beneficiarios de la herramienta. Por último, es uno de los lenguajes de programación más utilizados en el mundo por los desarrolladores de software, lo que da una prueba fiable de la popularidad y utilidad de este lenguaje para el desarrollo de aplicaciones informáticas que abarcan un número significativo de escenarios (Oracle Corporation).

1.3.3 Lenguaje de modelado

Para el desarrollo de la aplicación es necesario seleccionar un lenguaje de modelado que permita una visión constructiva del software y que se encuentre en correspondencia con la metodología de desarrollo a utilizar.

UML 2.1

Entre los lenguajes de modelado visual más conocidos se encuentra el Lenguaje de Modelado Unificado (UML, en inglés *Unified Model Language*), el cual se usa para entender, diseñar, hojear, configurar, mantener y controlar la información sobre un sistema de software. El lenguaje de modelado pretende

unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. UML incluye conceptos semánticos, notación y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Captura decisiones y conocimientos sobre los sistemas que se deben construir. Con la utilización de UML es posible llevar a cabo las siguientes funciones (Jacobson, 2000):

Visualizar: permite expresar de forma gráfica un sistema de forma que otro lo pueda entender.

Especificar: permite especificar cuáles son las características de un sistema antes de su construcción.

Construir: a partir de los modelos especificados se pueden construir los sistemas diseñados.

Documentar: los propios elementos gráficos sirven como documentación del sistema desarrollado y pueden servir para su futura revisión.

Conocidas las funciones que trae consigo el empleo de un lenguaje de modelado se decidió utilizar UML 2.1 para facilitar al programador la implementación del sistema una vez que sea diseñado.

1.3.4 Herramienta CASE

Ingeniería de Software Asistida por Computadora (del inglés Computer Aided Software Engineering-CASE) es un tipo de ingeniería de software en la que se intenta aumentar la eficacia de sus procesos, al soportar la realización de las tareas con el uso de tecnologías. Las herramientas CASE son usadas en algunas de las fases de desarrollo de sistemas de información, incluyendo análisis, diseño e implementación. Tienen como objetivo fundamental proveer un lenguaje para describir el sistema general, que sea lo más explícito posible a la hora de generar determinados programas.

Visual Paradigm para UML v8.0

Visual Paradigm para UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Este software ayuda a una más rápida construcción de aplicaciones de calidad y a un menor costo (MORA., 2007). Algunas de sus características son:

1. Disponibilidad en múltiples plataformas (Windows, Linux).
2. Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
3. Capacidades de ingeniería directa e inversa.
4. Modelo y código que permanece sincronizado en todo el ciclo de desarrollo
5. Disponibilidad de múltiples versiones, con diferentes especificaciones.
6. Licencia: gratuita y comercial.
7. Generación de código para Java y exportación como HTML.
8. Importación y exportación de ficheros XMI.

Se decidió usar Visual Paradigm para UML en su versión 8.0 debido a que es la herramienta CASE estándar que utiliza la Universidad en proyectos de gran alcance y al ser multiplataforma puede utilizarse en diferentes sistemas operativos. Brinda la posibilidad de modelar una gran variedad de diagramas, facilitando la codificación desde diagramas, la organización y el entendimiento por parte de los desarrolladores. Además, es una herramienta libre y contribuye a lograr mayor rapidez en la construcción de aplicaciones informática.

1.3.5 Entorno de desarrollo integrado (IDE).

Un Entorno de Desarrollo Integrado (del inglés, Integrated Development Environment, IDE, por sus siglas en inglés) es un programa compuesto por una serie de herramientas que utilizan los desarrolladores para escribir el código. Puede soportar varios lenguajes de programación o puede estar diseñado para uno únicamente. (Viñolo, 2012.)

NetBeans

El IDE NetBeans es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software. Este IDE es gratuito, y de código abierto. El mismo tiene gran éxito con una gran base de usuarios y una comunidad en constante crecimiento (TIM BOUDREAU, 2002).

Se decidió usar el NetBeans porque es gratuito, de fácil uso y ayuda a la flexibilidad en el trabajo por sus posibilidades de modificación por parte de los desarrolladores. Además, se quiere lograr un sistema

multiplataforma y esta herramienta permite crear aplicaciones multiplataforma. Su uso permite agilizar el trabajo del equipo de desarrollo, pues posee facilidades para el completamiento de código y propone un esqueleto para organizarlo. También, admite que las aplicaciones creadas por el IDE se adhieran a los estándares de la industria y proporciona facilidades y garantías para la migración lo cual es una gran ventaja para futuros cambios.

1.3.6 Sistema Gestor de Base de Datos (SGBD).

Un SGBD permite controlar el acceso a los datos, asegurar su integridad, gestionar el acceso concurrente a ellos, recuperar los datos tras un fallo del sistema y hacer copias de seguridad. Las bases de datos y los sistemas para su gestión son esenciales para cualquier área de negocio, y deben gestionarse con esmero.

SQLite es un sistema de gestión de bases de datos relacional, el cual es un proyecto de dominio público. A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción. En su versión 3, SQLite permite bases de datos de hasta 2 Terabytes de tamaño. Algunas de sus características más atractivas e importantes son (SQLite, 2011):

- Tamaño: SQLite tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.

- Rendimiento de base de datos: SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.
- Portabilidad: se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.

- Estabilidad: SQLite es compatible con ACID, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad.
- SQL: implementa un gran subconjunto de la ANSI – 92 SQL[7] estándar, incluyendo sub-consultas, generación de usuarios, vistas y triggers.
- Interfaces: cuenta con diferentes interfaces del API, las cuales permiten trabajar con Java, C++, PHP, Python, Ruby.
- Costo: SQLite es de dominio público y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente.

Con el uso de SQlite se garantiza tener autocontenida dentro de la misma aplicación la base de datos que incluye las plantillas de formularios y encuestas necesarias para realizar el proceso de captura de datos, así como los formularios y encuestas. Además al ser SQLite independiente, ya que no se comunica con un motor de base de datos, sino que las librerías de SQLite pasan a integrar la aplicación, se pueden gestionar bases de datos sin necesidad de tener instalado o conectar con un servidor de base de datos. Es por esto que para un entorno de trabajo con falta de conectividad, donde se requiere de aplicaciones portables, SQLite es el sistema gestor de base de datos que mejor se ajusta.

1.3.7 Administrador gráfico de base de datos.

Como administrador gráfico para SQlite se decide utilizar SQLiteman o también conocido como SQLite Manager, pues esta herramienta es de dominio público y es muy eficiente. Tiene una interfaz muy clara, dividida en pestañas para los elementos de diseño, administración y la elaboración de instrucciones SQL, con la posibilidad de crear y navegar por las tablas, índices y vistas, insertar, eliminar y editar las tablas, ejecutar sentencias SQL entre otros mecanismos. Dispone de un completo sistema de generación de informes, exportables en una gran variedad de formatos, incluyendo HTML, CSV y XML. En definitiva, SQLite Manager ofrece una forma más amena para navegar entre los objetos de la base de datos, gestionar la base de datos y construir instrucciones SQL, junto a una tabla completa con los resultados de las peticiones formuladas (Dossier, 2014).

[7] ANSI – 92 SQL: versión mejorada del estándar "SQL-86" o "SQL1"del lenguaje SQL; Instituto Nacional Estadounidense de Estándares (ANSI).

Conclusiones del capítulo: YAAA

En el presente capítulo se analizaron los conceptos asociados al objeto de estudio y las herramientas que se emplearán en la solución, mostrando como resultado las características claves que servirán de base para el diseño de la propuesta de trabajo. Se seleccionó como metodología para guiar todo el proceso de desarrollo OpenUp; UML v2.0 como lenguaje de modelado para visión constructiva del software; Visual Paradigm para UML v8.0 como herramienta CASE para describir el sistema lo más detallado posible a la hora de generar la solución; Java como lenguaje de programación; NetBeans v7.3.1 como IDE de desarrollo; SQLite como Sistema Gestor de Base de Datos para controlar el acceso a los datos y SQLiteman como administrador gráfico de base de datos.

Capítulo 2: Análisis y diseño de la aplicación

En este capítulo se describen los elementos asociados al análisis y diseño de la herramienta a desarrollar. A partir del modelo de dominio, se detalla la propuesta de solución para el problema de la investigación. Se obtienen los requisitos funcionales y no funcionales a tener en cuenta para la construcción del sistema. Se identifican los actores, casos de uso y las relaciones existentes entre ellos. Además se diseñan el diagrama de clases y el diagrama de casos de uso haciendo uso de patrones.

2.1 Modelo de dominio del negocio

El modelo dominio es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Este facilita a través de un vocabulario común que ayude a usuarios, clientes, desarrolladores e interesados a comprender el argumento principal del sistema (Díaz Anton, 2003). La figura 1 muestra el modelo de dominio:

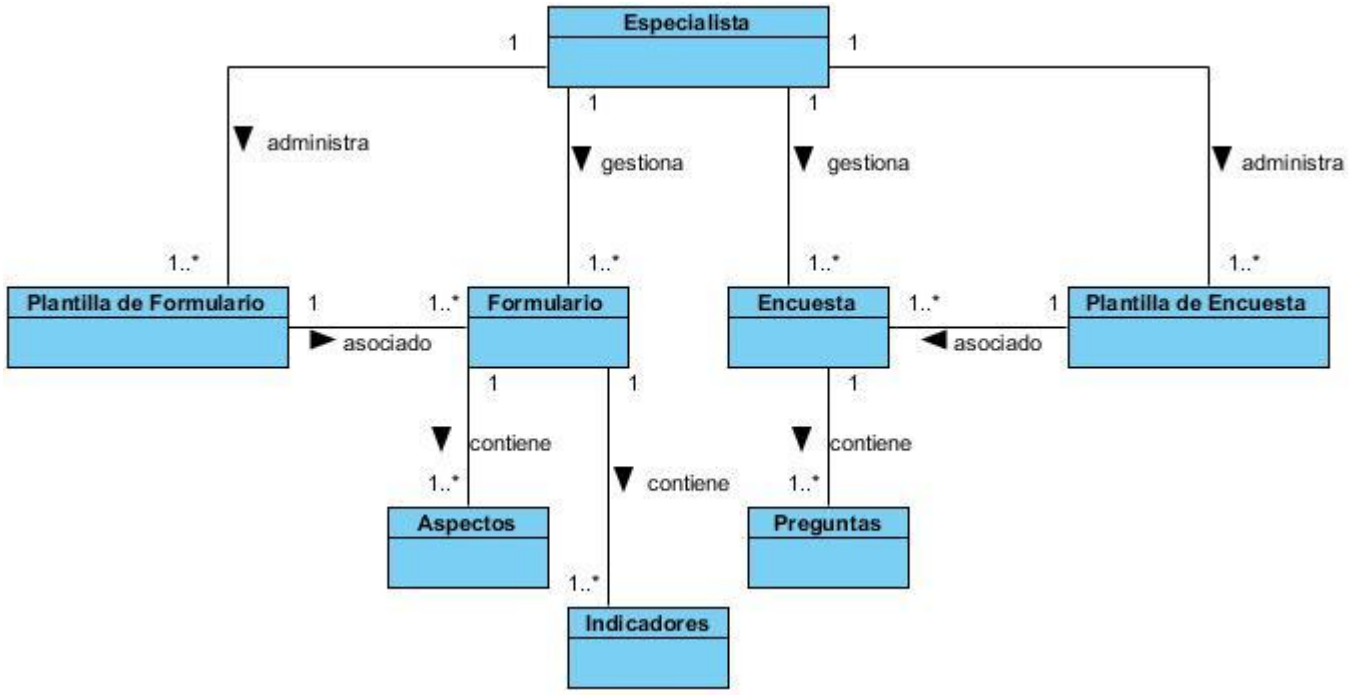


Figura 1. Diagrama del Modelo del Dominio

- **Descripción de las clases del dominio:**

Especialista: es el usuario que interactúa con el sistema, encargado de gestionar los formularios y las encuestas así como las plantillas de formularios y las plantillas de encuesta, la inserción, eliminación y actualización de los mismos. Este usuario tiene la tarea de digitalizar la información estadística. Él debe cargar los formularios y encuestas, llenarlos, verificar que los datos entrados estén correctos y luego guardarlos.

Encuesta: una encuesta está conformada por: las bases metodológicas, por el encabezado que constituye un tipo especial de sección, y preguntas que a su vez contienen las variables. La encuesta tiene un ciclo de vida, inicialmente en el estado no activado se aplica el proceso de diseño de la encuesta, luego de aprobado el diseño, esta pasa a estado activado donde se pone en marcha al proceso de pre visualización de la encuesta, se publica y se le da un seguimiento mientras se encuentra activada la misma.

Formulario: un formulario está conformado por: el encabezado del formulario, cuerpo del formulario y herramienta del captador. El formulario tiene un ciclo de vida, inicialmente es enviado a digitalización, luego a validación y finalmente a archivo.

Plantilla de Encuesta: estructura o modelo definido de una encuesta, el cual es seleccionado por el digitalizador y llenado según sus campos.

Plantilla de Formulario: estructura o modelo definido de un formulario, el cual es seleccionado por el digitalizador y llenado según sus campos.

2.2 Requisitos del sistema

Los requisitos del sistema no son más que esa condición que debe ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Se denomina como: condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo, define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen (Autores, 2009).

Partiendo de la descripción de los conceptos más importantes del sistema representados en el Modelo de Dominio, se determinaron los requisitos funcionales y no funcionales del sistema.

2.2.1 Requisitos Funcionales (RF) del sistema

Los RF son capacidades o condiciones que el sistema debe cumplir. Expresan una especificación detallada de las responsabilidades del sistema en cuestión y permiten determinar de una manera clara, lo que debe hacer el sistema (Pressman, 2005.).

Se identificaron los siguientes RF:

Tabla 2. Requisitos Funcionales (RF) del sistema.

RF1. Importar plantilla de tipo formulario.	
Descripción	El sistema debe permitir importar solo desde un archivo XML una plantilla de formulario.
RF2. Importar plantilla de tipo encuesta.	
Descripción	El sistema debe permitir importar solo desde un archivo XML una plantilla de encuesta.
RF3. Exportar plantilla de tipo formulario a un archivo XML.	
Descripción	El sistema debe permitir exportar la plantilla de tipo formulario a un archivo en formato XML compatible con SIGE.
RF4. Exportar plantilla de tipo encuesta a un archivo XML.	
Descripción	El sistema debe permitir exportar la plantilla de tipo encuesta a un archivo en formato XML compatible con SIGE.
RF5. Buscar plantilla.	
Descripción	El sistema debe permitir la búsqueda de una plantilla mediante uno de los siguientes aspectos: Número, Subnúmero, Identidad, Clasificación o Tipo.
RF6. Buscar plantilla de forma avanzada.	
Descripción	El sistema debe permitir la búsqueda de una plantilla mediante los aspectos: Número, Subnúmero, Identidad, Clasificación y Tipo.
RF7. Eliminar plantilla de tipo formulario.	
Descripción	El sistema debe permitir eliminar una plantilla de formulario previamente seleccionada.

RF8. Eliminar plantilla de tipo encuesta.	
Descripción	El sistema debe permitir eliminar una plantilla de encuesta previamente seleccionada.
RF9. Captar datos en plantillas de tipo formulario.	
Descripción	El sistema debe permitir captar datos en una plantilla de tipo formulario.
RF10. Captar datos en plantillas de tipo encuesta.	
Descripción	El sistema debe permitir captar datos en una plantilla de tipo encuesta.
RF11. Listar plantillas.	
Descripción	El sistema debe mostrar las plantillas existentes de forma paginada.
RF12. Importar formulario desde un archivo en formato XML.	
Descripción	El sistema debe permitir importar solo desde un archivo XML un formulario.
RF13. Exportar un formulario hacia un archivo en formato XML.	
Descripción	El sistema debe permitir exportar un formulario hacia un archivo XML.
RF14. Exportar un formulario hacia un archivo en formato ZIP.	
Descripción	El sistema debe permitir exportar varios formularios en formato XML hacia un archivo ZIP.
RF15. Exportar un formulario hacia un archivo en formato MSET de tipo Microset NT.	
Descripción	El sistema debe permitir exportar los datos de un formulario hacia un archivo en formato Microset NT.
RF16. Exportar un formulario hacia un archivo en formato MSET de tipo Microset NT (DPA).	
Descripción	El sistema debe permitir exportar los datos de varios formularios hacia un archivo en formato Microset NT.
RF17. Exportar un formulario hacia un archivo en formato MSET de tipo Estados Financieros.	
Descripción	El sistema debe permitir exportar los datos de un formulario hacia un archivo en formato Estados Financieros.

RF18. Exportar un formulario hacia un archivo en formato MSET de tipo Estados Financieros (DPA).	
Descripción	El sistema debe permitir exportar los datos de varios formularios hacia un archivo en formato Estados Financieros.
RF19. Exportar un formulario hacia un archivo en formato CSV (Formulario).	
Descripción	El sistema debe permitir exportar los datos de un formulario hacia un archivo en formato CSV.
RF20. Exportar un formulario hacia un archivo en formato CSV (DPA).	
Descripción	El sistema debe permitir exportar los datos de formularios hacia un archivo en formato CSV.
RF21. Eliminar un formulario.	
Descripción	El sistema debe permitir eliminar un formulario previamente seleccionado.
RF22. Buscar formulario.	
Descripción	El sistema debe permitir la búsqueda de un formulario mediante uno de los siguientes aspectos: Número, Subnúmero, Informe Acumulado, Variante, Identidad, Unidad de Observación, Tipo y Protegido.
RF23. Buscar formulario de forma avanzada.	
Descripción	El sistema debe permitir la búsqueda de un formulario mediante los aspectos: Número, Subnúmero, Informe Acumulado, Variante, Identidad, Unidad de Observación, Tipo y Protegido.
RF24. Captar formulario parcialmente completado.	
Descripción	El sistema debe permitir la captación de datos en un formulario parcialmente completado.
RF25. Configurar la captación rápida de un formulario.	
Descripción	El sistema debe permitir configurar la captación rápida de formularios.
RF26. Visualizar formulario.	
Descripción	El sistema debe permitir visualizar un formulario.
RF27. Exportar formulario a formato PDF.	

Descripción	El sistema debe permitir exportar un formulario a formato PDF.
RF28. Visualizar nota metodológica de un formulario.	
Descripción	El sistema debe permitir visualizar la nota metodológica de un formulario.
RF29. Guardar formulario.	
Descripción	El sistema debe permitir guardar un formulario.
RF30. Nuevo formulario.	
Descripción	El sistema debe permitir crear un nuevo formulario a partir de una plantilla de formulario previamente captada.
RF31. Listar formularios.	
Descripción	El sistema debe mostrar los formularios existentes de forma paginada.
RF32. Importar encuesta desde un archivo en formato XML.	
Descripción	El sistema debe permitir importar solo desde un archivo XML una encuesta.
RF33. Exportar una encuesta hacia un archivo en formato XML.	
Descripción	El sistema debe permitir exportar una encuesta hacia un archivo XML.
RF34. Exportar una encuesta a formato ZIP.	
Descripción	El sistema debe permitir exportar una o varias encuestas hacia un archivo en formato ZIP.
RF35. Eliminar encuesta.	
Descripción	El sistema debe permitir eliminar una encuesta previamente seleccionada.
RF36. Buscar encuesta.	
Descripción	El sistema debe permitir la búsqueda de una encuesta mediante uno de los siguientes aspectos: Número, Subnúmero, Informe Acumulado, Variante, Identidad, Unidad de Observación, Tipo y Protegido.
RF37. Buscar encuesta de forma avanzada.	
Descripción	El sistema debe permitir realizar una búsqueda de las encuestas existentes según

	los siguientes parámetros: Número y Subnúmero, Unidad de Observación, Protegido, Observaciones y Fecha de Confección.
RF38. Captar encuesta parcialmente completada.	
Descripción	El sistema debe permitir captar los datos de una encuesta parcialmente completada.
RF39. Visualizar nota metodológica de una encuesta.	
Descripción	El sistema debe permitir visualizar la nota metodológica de una encuesta.
RF40. Guardar encuesta.	
Descripción	El sistema debe permitir guardar una encuesta.
RF41. Nueva encuesta.	
Descripción	El sistema debe permitir crear una encuesta nueva a partir de una plantilla de encuesta previamente captada.
RF42. Listar encuestas.	
Descripción	El sistema debe mostrar las encuestas existentes de forma paginada.
RF43. Captar pregunta de tipo "Campo de Texto".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Campo de texto sean captadas.
RF44. Captar pregunta de tipo "Caja de Texto".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Caja de Texto sean captadas.
RF45. Captar pregunta de tipo "Campo Numérico".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Campo Numérico sean captadas.
RF46. Captar pregunta de tipo "Campo de Fecha".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Campo de Fecha sean captadas.

RF47. Captar pregunta de tipo "Campo de Tiempo".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Campo de Tiempo sean captadas.
RF48. Captar pregunta de tipo "Selección Simple".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Selección Simple sean captadas.
RF49. Captar pregunta de tipo "Selección Múltiple".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Selección Múltiple sean captadas.
RF50. Captar pregunta de tipo "Filtro".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Filtro sean captadas.
RF51. Captar pregunta de tipo "Campo Tabla".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Campo Tabla sean captadas.
RF52. Captar pregunta de tipo "Dicotónica".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Dicotónica sean captadas.
RF53. Captar pregunta de tipo "Orden de Clasificación".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Orden de Clasificación sean captadas.
RF54. Captar pregunta de tipo "Matriz Evaluativa".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Matriz Evaluativa sean captadas.
RF55. Captar pregunta de tipo "Enlace".	
Descripción	El sistema debe permitir que las encuestas, con tipo de pregunta Enlace sean

	captadas.
RF56. Definir fecha del informe acumulado.	
Descripción	El sistema debe permitir establecer de forma permanente una fecha para informe acumulado.
RF57. Adicionar unidad de observación.	
Descripción	El sistema debe permitir adicionar una unidad de observación.
RF58. Modificar unidad de observación.	
Descripción	El sistema debe permitir modificar los datos de una unidad de observación.
RF59. Eliminar unidad de observación.	
Descripción	El sistema debe permitir eliminar una unidad de observación previamente seleccionada.
RF60. Listar unidades de observación.	
Descripción	El sistema debe mostrar las unidades de observación existentes.
RF61. Cambiar cantidad de elementos a mostrar en las listas.	
Descripción	El sistema debe permitir listar la cantidad especificada por el usuario en listas del sistema, dígase plantillas, formularios y encuestas.

2.2.2 Requisitos no Funcionales (RnF) del sistema

Los RnF describen las características, cualidades o propiedades que el producto debe tener, aunque no formen parte de sus funciones (Pressman, 2005.).

Para el desarrollo de la solución se identificaron los siguientes requisitos no funcionales**:**

Usabilidad

RnF1: El sistema debe permitir conocer al usuario las acciones que puede realizar, a partir de íconos sugerentes, alternativa textual o algún otro elemento que le facilite la orientación al usuario para el trabajo con el mismo.

Software

RnF2: Es necesaria la instalación de la máquina virtual de java para la ejecución del sistema.

Hardware

RnF3: Hardware requerido para desplegar y utilizar la aplicación**:** Ordenador Pentium III o superior, con 1.7 GHz de velocidad de microprocesador, con memoria RAM mínimo 256 MB, capacidad de espacio libre disco duro como mínimo de 2 GB.

Portabilidad

RnF4: El sistema será multiplataforma, razón por la cual podrá utilizarse en cualquier sistema operativo y arquitectura de hardware.

Requisitos Legales, de Derecho de Autor y otros

RnF5: Como producto, se distribuirá amparado bajo las normativas legales establecidas en el registro comercial emitido por las entidades jurídicas de la Universidad de las Ciencias Informáticas.

RnF6: El sistema contará con una ayuda la cual se entregará en formato PDF y en idioma Español.

2.3 Modelado de los Casos de Usos del Sistema (CUS)

El diagrama de CUS documenta el comportamiento de un sistema desde el punto de vista del usuario. Representa las relaciones existentes entre actores y casos de uso. Partiendo de los requisitos funcionales identificados se modeló el Diagrama de CUS, el cual se muestra en la figura 2:

.

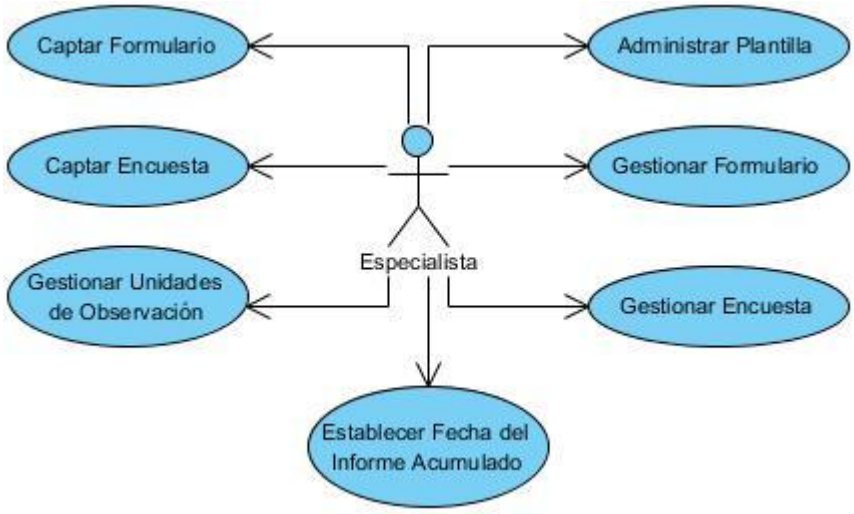


Figura 2. Diagrama de Casos de Uso del Sistema.

Para la creación del diagrama de CUS se tuvo en cuenta la utilización de los siguientes patrones de casos de uso:

- **CRUD parcial**, en el CU Administrar Plantillas pues la aplicación no permite operaciones de modificar y crear sobre las plantillas registradas en el sistema.

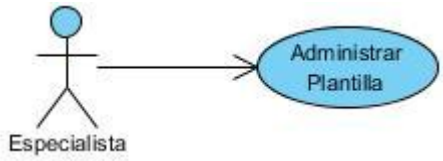


Figura 3. Evidencia el patrón CRUD parcial

- **CRUD total,** en los CU Gestionar Formulario, Gestionar Encuesta y Gestionar Unidades de Observación pues el sistema permite crear, eliminar, modificar y buscar los formularios, encuestas y las unidades de observación.

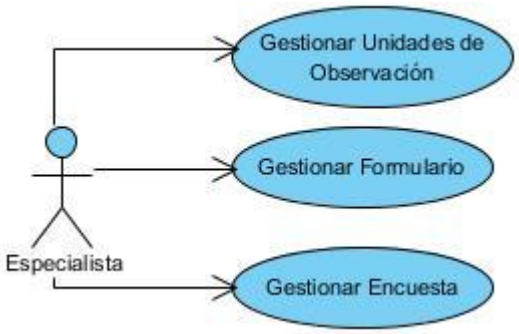


Figura 4. Evidencia el patrón CRUD total

2.3.1 Matriz de trazabilidad

La trazabilidad es la capacidad de describir y de seguir la vida de un requisito, en dirección hacia adelante y hacia atrás. A continuación se muestra una matriz de trazabilidad para representar a qué CU pertenece cada RF:

Tabla 3. Matriz de trazabilidad

RF/CU	Administrar Plantilla	Gestionar Formulario	Captar Formulario	Gestionar Encuesta	Captar Encuesta	Gestionar Unidades de Observación	Definir Fecha del Informe Acumulado
RF1-RF11, RF 61	X						
RF12-RF24, RF 61		X					
RF24-RF31			X				
RF32-RF38, RF 61				X			
RF38-RF55					X		
RF56-RF59						X	
RF60							X

2.3.2 Descripción de los actores

Tabla 4. Descripción de actores del sistema

Actor	Descripción
Especialista	Es el usuario que interactúa con el sistema, encargado de gestionar los formularios y las encuestas así como las plantillas de formularios y las plantillas de encuesta, la inserción, eliminación y actualización de los mismos. Este usuario tiene la tarea de digitalizar la información estadística. Él debe cargar los formularios y encuestas, llenarlos y luego guardarlos.

2.3.3 Descripción de los casos de usos del sistema

CU. Captar Formulario

Tabla 5. Descripción del CU Captar Formulario

Caso de Uso:	Captar Formulario
Actor:	Especialista
Resumen:	El caso de uso inicia cuando el Especialista selecciona una plantilla de formulario o un formulario parcialmente digitalizado y el botón captar. Luego completa con datos estadísticos los campos requeridos. El caso de uso termina cuando el especialista selecciona la opción **Guardar.**
Precondiciones:	El sistema tiene que estar conectado a la base de datos, tiene que haber seleccionado una plantilla de formulario o un formulario parcialmente digitalizado.
Referencias	RF24, RF25, RF26, RF27, RF28, RF29, RF30, RF31.
Complejidad	Alta
Flujo Normal de Eventos	
Sección “Selección de plantilla o formulario”	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción **Captar**. ➢ Captar de forma rápida formularios a partir de una plantilla de formulario, ver sección **Captación rápida de formularios**. ➢ Captar de forma rápida formularios parcialmente completados, ver sección **Captación rápida de formularios**	

<table>
<tr><td colspan="2">parcialmente completados.</td><td></td></tr>
<tr><td colspan="3">Sección “Captación rápida de formularios”</td></tr>
<tr><td>Acción del Actor</td><td colspan="2">Respuesta del Sistema</td></tr>
<tr><td></td><td colspan="2">1. El sistema muestra los elementos del formulario:
➢ Captación rápida que incluye: Encabezado del formulario y Cuerpo del Formulario.
➢ Nota Metodológica. Ver sección “Nota Metodológica”.</td></tr>
<tr><td>2. El especialista llena los campos del encabezado (Acumulado hasta, variante, Unidad de Observación y Observaciones) y selecciona la opción Guardar para comenzar la digitación.

4. El especialista comienza la digitalización llenando los campos del Cuerpo del Formulario y selecciona la opción Guardar.

6. El especialista selecciona la opción Cerrar.</td><td colspan="2">3. El sistema guarda el formulario en la base de datos y actualiza la lista de formularios. El sistema muestra el mensaje “Formulario guardado satisfactoriamente” y habilita las opciones:
➢ Formato. Ver sección “Formato”.
➢ Configuración. Ver sección “Configuración”.
➢ Nuevo. Ver sección “Nuevo”.

5. El sistema guarda los datos del formulario en la base de datos y muestra el mensaje “Recurso guardado satisfactoriamente”.

7. El sistema cierra el formulario. Finaliza el CU.</td></tr>
<tr><td colspan="3">Flujo Alterno al paso 3 “Datos incorrectos y/o campos vacíos”</td></tr>
<tr><td colspan="2">Acción del Actor</td><td>Respuesta del Sistema</td></tr>
<tr><td colspan="2"></td><td>3a. El sistema muestra el mensaje “El código de variante seleccionado no es válido”.
3b. El sistema muestra el mensaje “Debe insertar una unidad de observación válida. Número de entre 5 y 15 dígitos”.</td></tr>
<tr><td colspan="3">Sección “Captación rápida de formularios parcialmente completados”</td></tr>
</table>

Acción del Actor	Respuesta del Sistema
	1. El sistema muestra los elementos del formulario: ➢ Captación rápida que incluye: Encabezado del formulario y Cuerpo del Formulario. ➢ Nota Metodológica. Ver sección **"Nota Metodológica".** ➢ Formato. Ver sección **"Formato".** ➢ Configuración. Ver sección **"Configuración".** ➢ Nuevo. Ver sección **"Nuevo".**
2. El especialista comienza la digitalización llenando los campos del Cuerpo del Formulario y selecciona la opción **Guardar**. 4. El especialista selecciona la opción **Cerrar**.	3. El sistema guarda los datos del formulario en la base de datos y muestra el mensaje "Recurso guardado satisfactoriamente". 5. El sistema cierra el formulario. Finaliza el CU.
Sección "Visualizar Nota Metodológica"	
Acción del Actor	Respuesta del Sistema
1. El especialista selecciona **Nota Metodológica** del formulario que está digitalizando.	2. El sistema muestra la Nota Metodológica en caso que el formulario la tenga.
Sección "Formato"	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción **Formato**.	2. Guarda el formulario y muestra la vista del formulario final, luego de ser digitalizado. Permite una vez que se visualiza el formato del formulario, abrir o guardar este hacia un archivo (Exportar a PDF).
2. Selecciona la opción **Exportar a PDF.**	4. Se muestra la ventana de descarga en la que se especifica, según la configuración del mismo, la localización en la que se guardará el archivo que se

5. Cierra el formulario en formato PDF.	genera como resultado de la exportación y se guarda. Una vez guardado se muestra una vista previa del formulario en el formato guardado. 5. Cierra el formulario en formato PDF y regresa al entorno de trabajo.
Sección “Configuración”	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción **Configurar**. 3. Selecciona los elementos a configurar. 4. El especialista selecciona la opción “Tipo de autocompletado” 7. El especialista selecciona una de las opciones y da clic en el botón **Aceptar** para comenzar la digitalización.	2. El sistema muestra la configuración dividida en dos elementos: ➢ Aspectos del Formulario. ➢ Tipos de Autocompletado. 4. El sistema activa las funciones de autocompletado para el aspecto seleccionado. 6. El sistema permite autocompletar con las siguientes opciones: • Igualar a Celda. • Formula. • Igualar a Literal. 8. El sistema muestra la plantilla con los cambios de autocompletado según la opción seleccionada por el especialista.
Sección “Nuevo”	
Acción del Actor	Respuesta del Sistema

1. Selecciona la opción **Nuevo**. 3. El especialista selecciona la opción **Si** para continuar.	2. El sistema muestra el mensaje “Los cambios no guardados se perderán ¿Desea guardar el formulario?”. 4. El sistema guarda los cambios del formulario y muestra un nuevo formulario asociado a esa plantilla.

Los restantes casos de uso del sistema se encuentran descritos en el expediente de proyecto.

2.4 Patrones

Los patrones constituyen una guía para el diseño del software. Su objetivo es la solución de problemas que ocurren repetidamente dentro de un contexto muy bien definido. Además deben ser reusables, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. Son de gran utilidad para describir las mejores prácticas, buenos diseños, y encapsulan la experiencia permitiendo su reutilización. Definen la estructura de un sistema de software, los cuales a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes del mismo sistema, con el objetivo de facilitar la tarea del diseño de tal sistema (Gamma, 1995).

2.5.1 Patrones arquitectónicos

Uno de los patrones más generalizados en el diseño es el Modelo-Vista-Controlador (MVC). Este patrón logra que la implementación de aplicaciones sea rápida y sencilla, representándolo a través de tres elementos fundamentales: modelo, vista y controlador.

Modelo: representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. La BD pertenece a esta capa.

Vista: transforma el modelo en una interfaz visual para la interacción del usuario con el sistema.

Controlador: se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

2.4.2 Patrones de diseño

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. Para el desarrollo de la solución se tuvieron en cuenta los patrones GRASP (Patrones Generales de Software para Asignación de Responsabilidades), debido a que describen los principios fundamentales de la asignación de responsabilidades a objetos expresados en forma de patrones. Se considera que más que patrones, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

- **GRASP:**

Experto: se utilizó el patrón Experto debido a que plantea que se debe asignar la responsabilidad al experto en información, que en este contexto sería la clase que cuenta con la información necesaria para cumplir la responsabilidad. Un ejemplo donde se evidencia es la clase TablaCaptar.

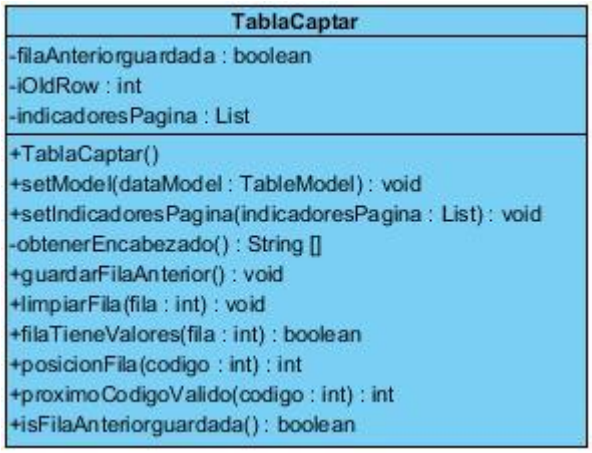


Figura 5: Ejemplo de la utilización del patrón Experto

Creador: se empleó el patrón Creador, el cual se encarga de asignar a una clase la responsabilidad de crear una instancia de otra. Es de gran importancia saber asignar la responsabilidad de crear instancias de una clase sólo a aquella que la requiera. Este patrón se evidencia en la clase Modelo DAO, pues estas clases son las que mediante instancias a las clases del paquete entidad permiten la comunicación entre la clase controladora y la base de datos.

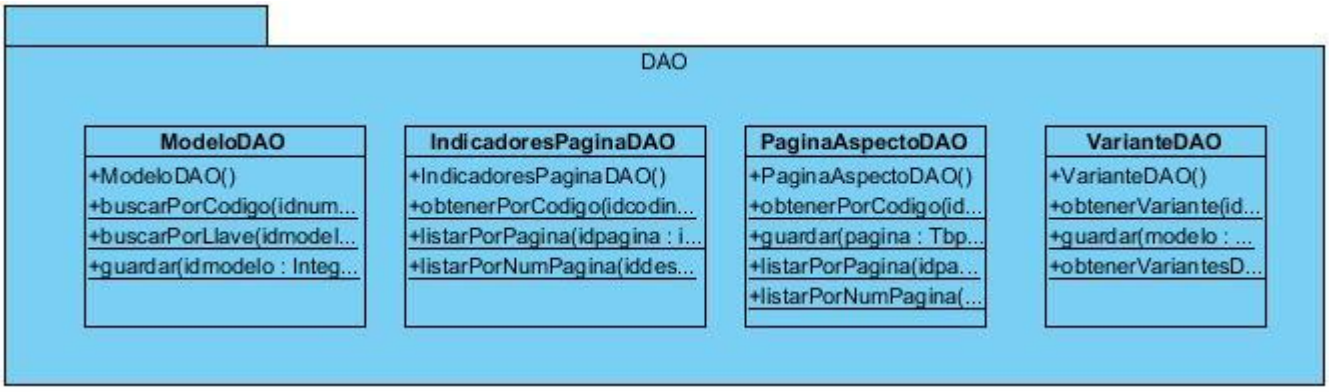


Figura 6. Ejemplo de la utilización del patrón Creador

Alta Cohesión: consiste en asignar una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es la medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Permite mejorar la claridad y la facilidad con que se entiende el diseño y genera a menudo un bajo acoplamiento.

Bajo Acoplamiento: consiste en asignar las responsabilidades de forma tal que las clases se comuniquen con el menor número de clases que sea posible. Lo cual disminuye la repercusión en las clases por la modificación de otra, potenciando la reutilización y disminuyendo la dependencia entre las clases.

Controlador: consiste en asignar la responsabilidad del manejo de mensajes de los eventos del sistema a una clase, o sea, esta clase tiene la responsabilidad de manejar los eventos del sistema. Generalmente los eventos del sistema son generados por actores externos. Un ejemplo donde se evidencia este patrón es en la clase controladora ControladorCaptar.

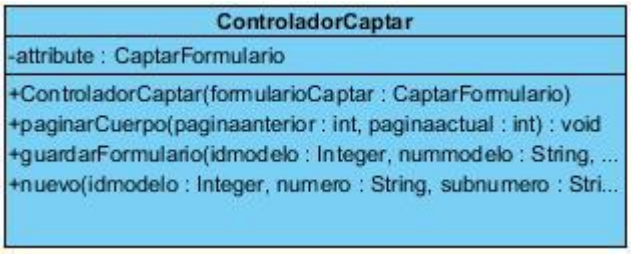


Figura 7. Ejemplo de la utilización del patrón Controlador

- **Patrón de diseño DAO (Data Access Object)**

El patrón DAO es una solución al problema del diferencial de impedancia entre un programa de aplicación orientada a objetos y una base de datos relacional, empleando únicamente la interface de programación (API) nativa del manejador de base de datos. (Autores, 2000)

Este patrón se utiliza para:

- Abstraer y encapsular los accesos.
- Gestionar las conexiones a la fuente de datos.
- Obtener los datos almacenados.

El uso de este patrón se evidencia en las clases del paquete DAO contenido dentro del paquete Modelo en la figura 6 y brinda las siguientes ventajas:

- Cualquier objeto no requiere conocimiento directo del destino final de la información que se manipula.
- Se baja el nivel de acoplamiento entre clases, reduciendo la complejidad de realizar cambios.
- Se aíslan las conexiones a la fuente de datos en una capa fácilmente identificable y mantenible.

2.5 Modelo del diseño

El modelo del diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, sirve de abstracción de la implementación del sistema y es, de ese modo, utilizado como una entrada fundamental de las actividades de implementación. En el modelo de diseño, los casos de uso son realizados por las clases de diseño y sus objetos (Jacobson, 2000).

2.4.1 Diagrama de clases del diseño

Un diagrama de clases del diseño es una representación concreta de los métodos y atributos de cada una de las clases del sistema que se debe implementar. Para mostrar de forma simple la colaboración y las tareas de cada una de ellas en relación al sistema que conforman. Estos diagramas representan la parte estática del sistema (Jacobson, y otros, 2000).

La figura 6 muestra el diagrama de clases del diseño del caso de uso Captar Formulario, integrado con el diagrama de paquetes para lograr una mejor comprensión de la arquitectura del sistema. En el paquete Vista se encuentran todas las clases que permiten la interacción del usuario con el sistema y recogen la

información necesaria para responder a sus peticiones. El paquete Controlador contiene la clase ControladorCaptar que responde a eventos (usualmente acciones del usuario) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre la información. Todas las clases necesarias para interactuar con la Base de Datos (BD) se encuentran en el paquete Modelo. Este contiene a su vez los paquetes DAO (Data Access Object) y Entidades. Las clases del paquete DAO suministran una interface común entre la aplicación y la BD utilizando el patrón DAO (Objeto de Acceso a Datos). En las clases del paquete Entidades se representan los datos del negocio que son almacenados en la base de datos del sistema.

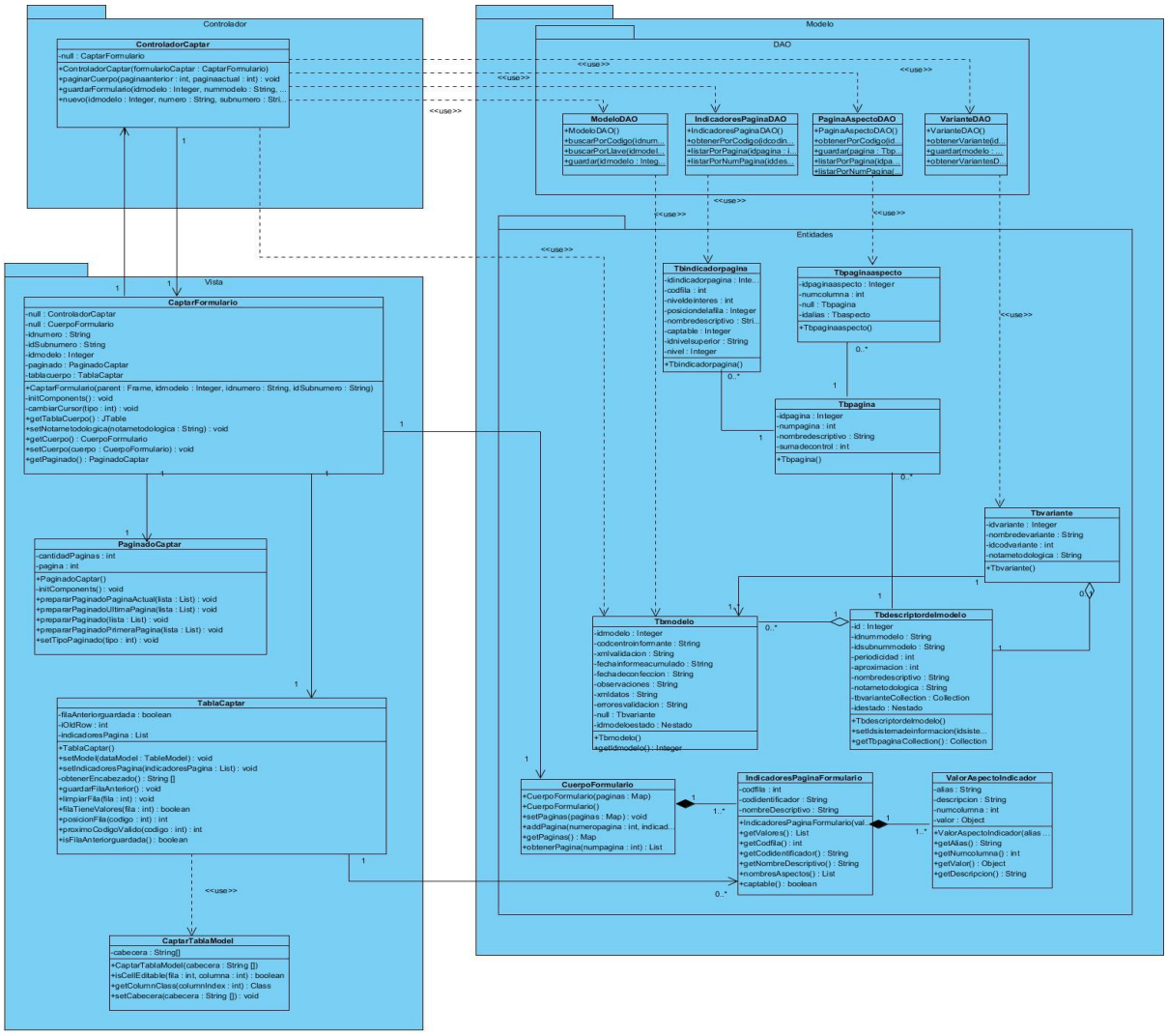


Figura 8. Diagrama de clases del CU Captar Formulario.

2.6 Diagrama de clases persistentes

La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Las clases persistentes constituyen información de larga duración, o sea, son los datos que necesitan almacenarse para gestionarse en cualquier momento. Una vez identificadas las clases cuyos estados son los más trascendentes en el tiempo de vida de la aplicación a desarrollar, se crea el diagrama de clases y se seleccionan las que constituyen una necesidad conservar su estado; quedando modelado el diagrama de clases persistentes.

Las principales características que posee este diagrama es describir la estructura del sistema mostrando sus clases, atributos y las relaciones entre ellas. Generalmente, estas clases tienen como origen las

clases clasificadas como entidad, porque modelan la información del sistema y el comportamiento asociado de algún fenómeno o concepto (Anaisa Hernández, 2002). La figura 10 muestra el diagrama de clases persistentes del CU Captar Formulario.

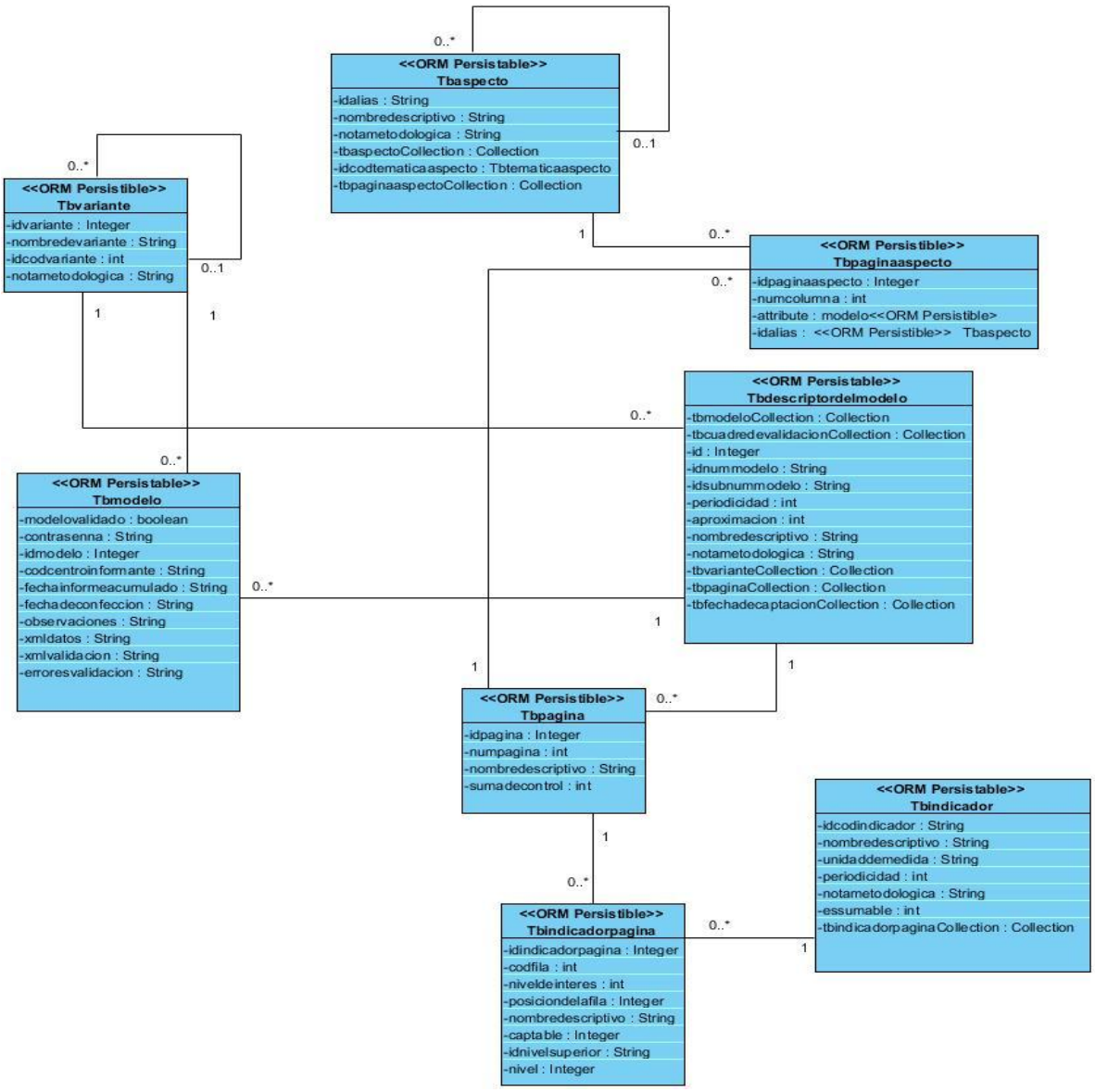


Figura 9. Diagrama de clases persistentes del CU Captar Formulario

2.7 Modelo de datos

Un modelo de datos es un sistema formal y abstracto que permite describir los datos de acuerdo con reglas y convenios predefinidos mediante una representación gráfica (Jacobson, 2000.). Permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan esos elementos entre sí, es un lenguaje utilizado para la descripción de una BD.

Dada la importancia que desempeña una base de datos en cualquier aplicación en la cual se gestione información, su diseño es un proceso esencial que permite el almacenamiento de la información de forma coherente y organizada. A continuación se muestran las entidades de la base de datos y las relaciones entre ellas.

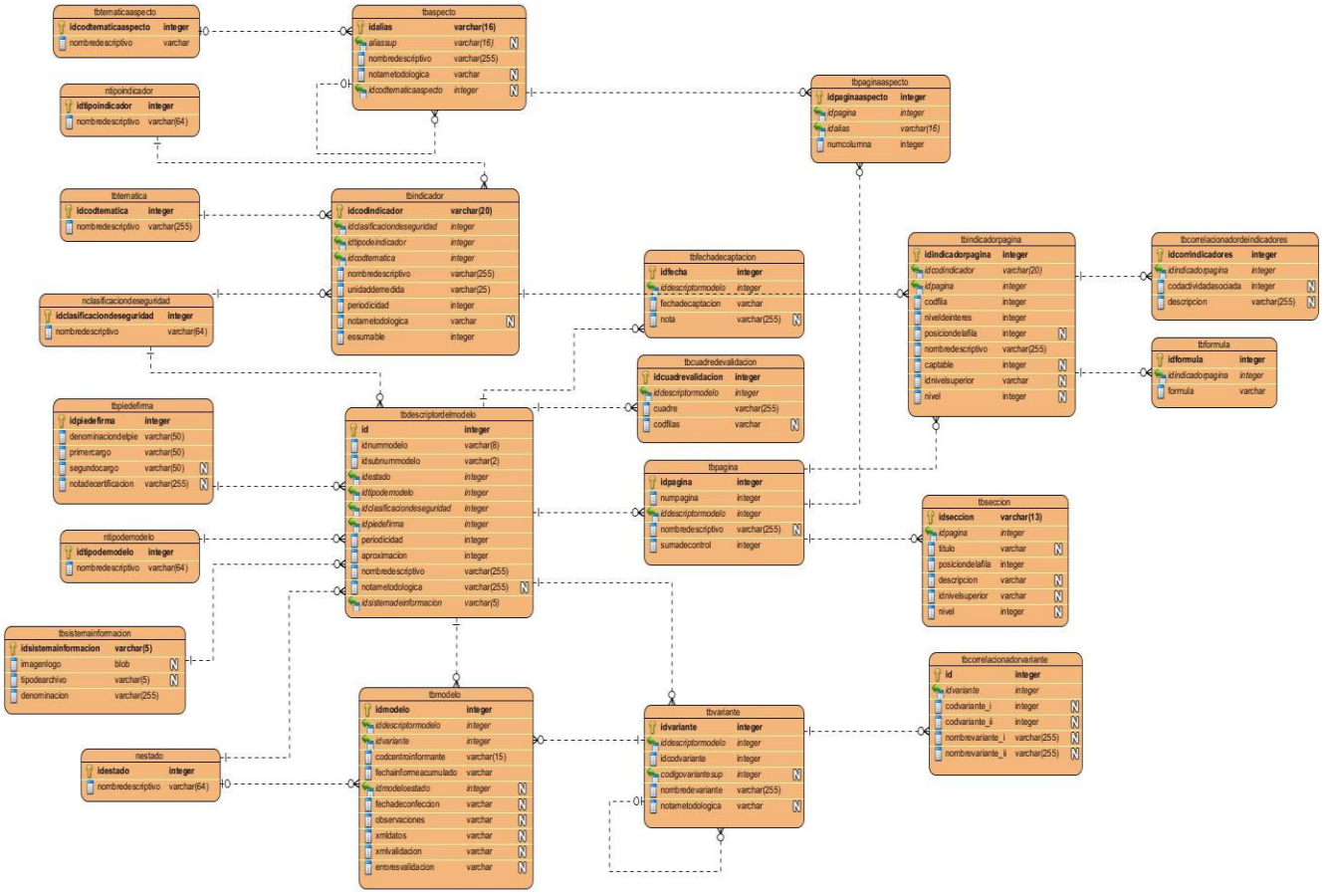


Figura 10. Diagrama Entidad Relación

Conclusiones del capítulo.

En el presente capítulo se obtuvieron: 61 requisitos funcionales y 8 no funcionales, el modelo de dominio, y el diagrama de casos de uso del sistema, este último con el apoyo de los patrones de casos de uso: CRUD total y CRUD parcial. Con la realización de los casos de uso, se creó el diagrama de clases del diseño integrado por paquetes haciendo uso del patrón arquitectónico MVC, para lograr una mejor comprensión de la arquitectura del sistema. El uso de los patrones de diseño GRAPS: Experto, Creador, Alta Cohesión, Bajo Acoplamiento y Controlador; junto al patrón DAO sirvieron de guía para la implementación del software. Se diseñó el diagrama de clases persistentes y se obtuvo el modelo de datos del sistema, quedando descrito la estructura de la base de datos.

Capítulo 3: Implementación y pruebas de la aplicación

En el presente capítulo se realiza una descripción de cómo los elementos del modelo de diseño se implementan en términos de componentes. Se analizan los artefactos principales del sistema como el modelo de implementación que incluye componentes, subsistemas de implementación y diagramas de componentes. Se llevarán a cabo las pruebas para validar el correcto funcionamiento de la herramienta.

3.1. Modelo de Implementación

El Modelo de Implementación describe cómo los elementos del modelo de diseño, se implementan en términos de componentes, ficheros de código fuente, ejecutables, entre otras. Describe también cómo se organizan los componentes de acuerdo a los mecanismos de estructuración disponibles en el entorno de implementación y lenguaje o lenguajes de implementación empleados y cómo dependen los componentes unos de otros (Jacobson y otros, 2000). Es de gran utilidad a la hora de implementar el sistema, pues facilita la organización del producto y lo hace más comprensible a los desarrolladores.

3.1.2 Diagrama de Componentes

Los diagramas de componentes representan cómo un sistema de software es dividido en componentes, mostrando las dependencias que existen entre ellos. Los componentes físicos incluyen: archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables y paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software, pero pueden usarse para modelar y documentar cualquier arquitectura de sistema; estos son utilizados para modelar la vista estática y dinámica de un sistema (Jacobson, y otros, 2000).

La figura 12 muestra el diagrama de componentes del CU Captar Formulario, donde se encuentran los paquetes de componentes y las relaciones entre éstos. El mismo está dividido en tres paquetes de implementación básicos:

- Modelo: agrupa los componentes que interactúan con la base de datos.
- Vista: agrupa los componentes que permiten la interacción del usuario con el sistema y recoge la información necesaria para responder a sus peticiones.

- Controlador: agrupa los componentes que se encargan de manejar el flujo central de las acciones del usuario con el sistema, apoyándose en el modelo para obtener la información necesaria para responder las peticiones de la vista.

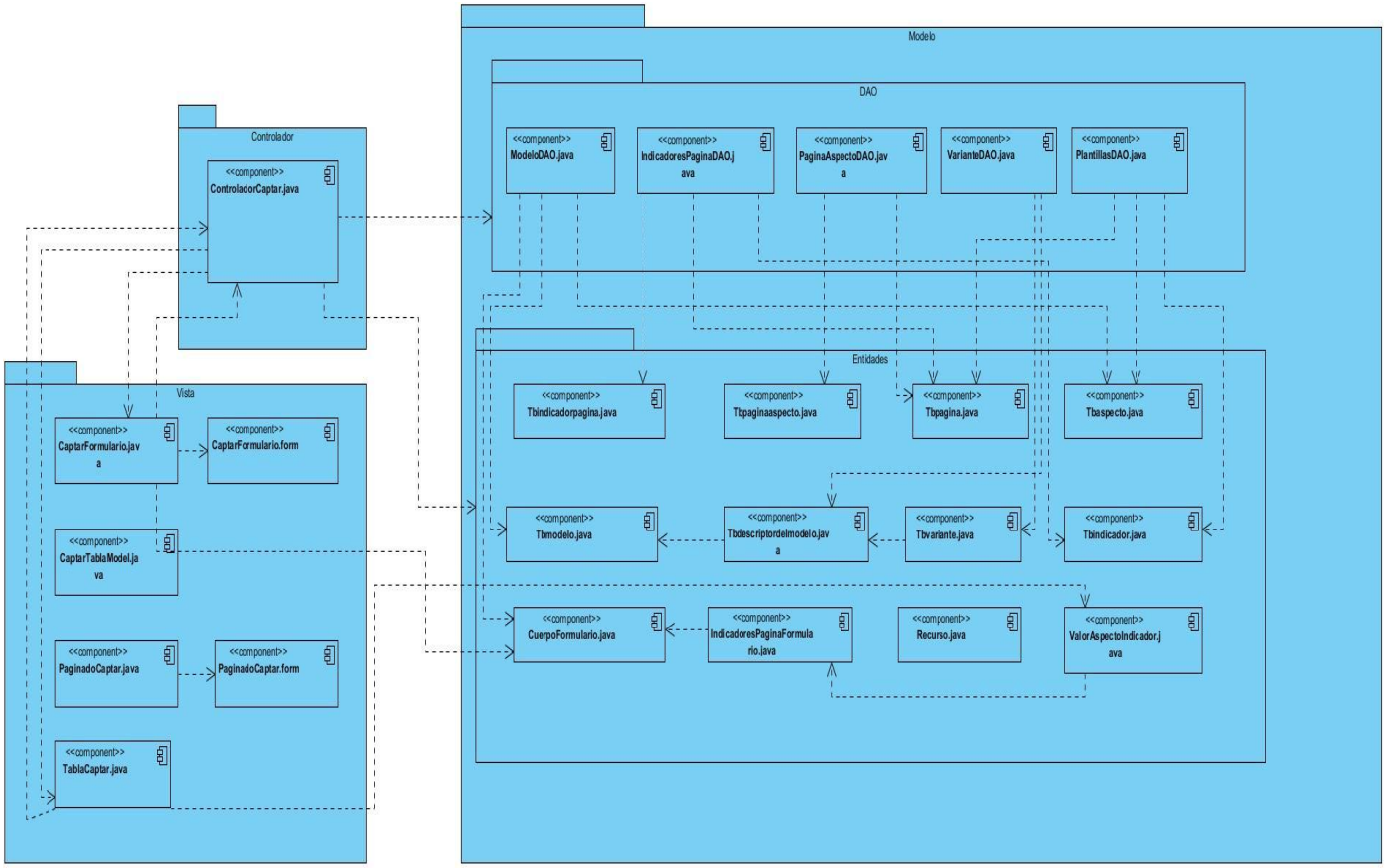


Figura 11. Diagrama de componentes de CU Captar Formulario

Es de señalar que la clase controladora se relaciona con todas las clases del paquete Modelo, pero para no sobrecargar el diagrama la relación que existe es entre la clase controladora y cada uno de los paquetes contenidos dentro del modelo.

3.1.3 Código Fuente

El código fuente es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar un programa. Por tanto, en el código fuente de un programa está descrito por

completo su funcionamiento. Estas instrucciones son escritas en un lenguaje de programación que consiste en un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura, el significado de sus elementos y expresiones. La Figura 13 muestra un fragmento de código del CU Captar Formulario.

3.1.4 Estándar 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 vistas a generar un código de alta calidad. Posibilita que un equipo de programadores mantenga un código de calidad sobre el que se efectuarán luego revisiones.

El estándar de codificación seleccionado para el desarrollo de la aplicación es el CamelCase:

CamelCase: es un estilo de escritura que se aplica a frases o palabras compuestas, dentro de este estilo existen dos tipos: **UpperCamelCase y lowerCamelCase.**

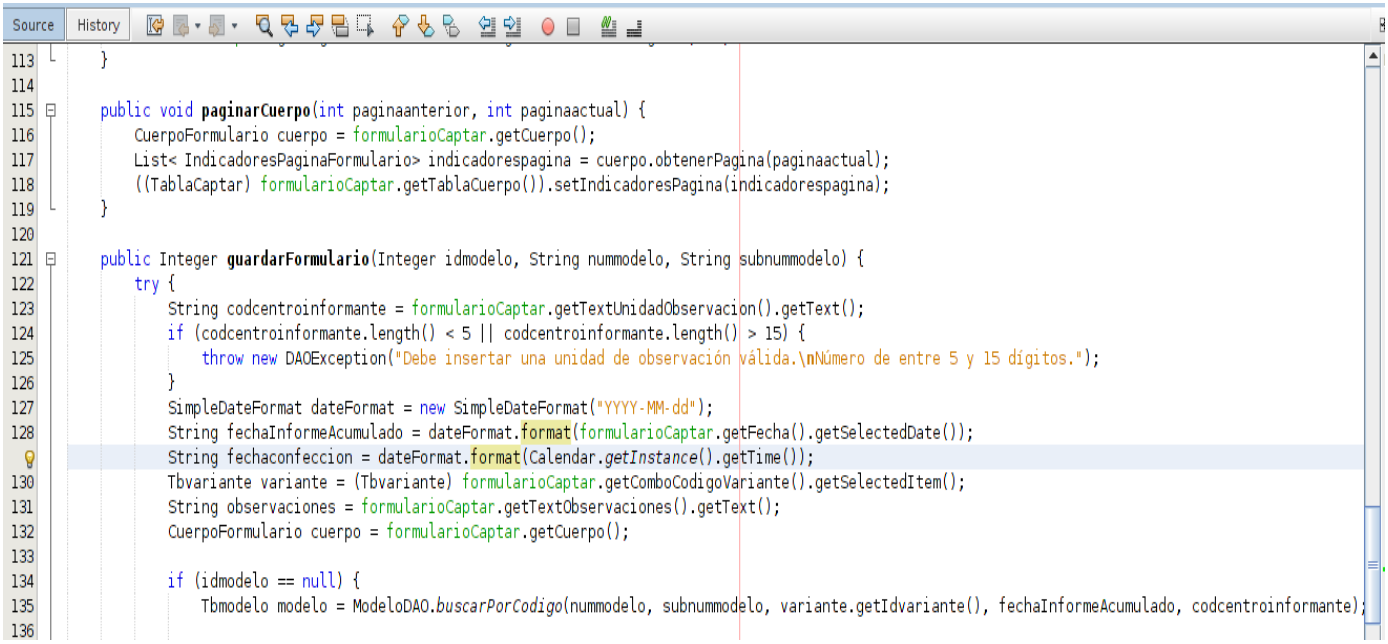
UpperCamelCase: las palabras que forman el nombre se escriben juntas y la primera letra de cada una de ellas en mayúscula.

lowerCamelCase: las palabras que forman el nombre se escriben juntas, la primera letra de la primera palabra en minúscula y del resto de las palabras, la primera letra en mayúscula.

A continuación se muestra el estilo de codificación utilizado y un fragmento de código fuente que lo evidencia:

- Todas las funciones y clases deben estar comentariadas. Los comentarios deben añadirse de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones o variables.
- Usar 1 línea en blanco para separar funciones.
- Nombrar las clases y funciones consistentemente, usar CamelCase de tipo lowerCamelCase para las funciones y upperCamelCase para los nombres de las clases.
- Usar indentación a 4 espacios, sangrías de 4 espacios.
- Espacios alrededor de operadores y luego de las comas.
- Todo bloque de sentencias entre llaves, aunque sea una sola sentencia después de un ***if.***

- El acceso/modificación de las propiedades de una clase (no constantes) siempre mediante métodos de acceso get/set.



```
113     }
114
115     public void paginarCuerpo(int paginaanterior, int paginaactual) {
116         CuerpoFormulario cuerpo = formularioCaptar.getCuerpo();
117         List< IndicadoresPaginaFormulario> indicadorespagina = cuerpo.obtenerPagina(paginaactual);
118         ((TablaCaptar) formularioCaptar.getTablaCuerpo()).setIndicadoresPagina(indicadorespagina);
119     }
120
121     public Integer guardarFormulario(Integer idmodelo, String nummodelo, String subnummodelo) {
122         try {
123             String codcentroinformante = formularioCaptar.getTextUnidadObservacion().getText();
124             if (codcentroinformante.length() < 5 || codcentroinformante.length() > 15) {
125                 throw new DAOException("Debe insertar una unidad de observación válida.\nNúmero de entre 5 y 15 dígitos.");
126             }
127             SimpleDateFormat dateFormat = new SimpleDateFormat("YYYY-MM-dd");
128             String fechaInformeAcumulado = dateFormat.format(formularioCaptar.getFecha().getSelectedDate());
            String fechaconfeccion = dateFormat.format(Calendar.getInstance().getTime());
130             Tbvariante variante = (Tbvariante) formularioCaptar.getComboCodigoVariante().getSelectedItem();
131             String observaciones = formularioCaptar.getTextObservaciones().getText();
132             CuerpoFormulario cuerpo = formularioCaptar.getCuerpo();
133
134             if (idmodelo == null) {
135                 Tbmodelo modelo = ModeloDAO.buscarPorCodigo(nummodelo, subnummodelo, variante.getIdvariante(), fechaInformeAcumulado, codcentroinformante);
136
```

Figura 12. Fragmento de código del CU Captar Formulario

3.2. Pruebas de software

La prueba de software consiste en ejecutar el programa con la intención de descubrir errores previos a la entrega al usuario final. Tiene como objetivo encontrar y documentar los defectos que pueden afectar la calidad del software. Las pruebas se aplican durante todo el ciclo de desarrollo del software para diferentes objetivos y en distintos niveles de trabajo (Pressman, 2005.).

Existen diferentes niveles de pruebas como: pruebas de desarrollador, independiente, unidad, integración, sistema y aceptación. Para comprobar que el sistema cumple con los requisitos previamente identificados se aplicarán las pruebas al nivel de desarrollador mediante el método de Caja Negra y prueba de aceptación.

Método 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 (Pressman, 2011).

Para evaluar las funcionalidades que ofrece el sistema se decide usar las pruebas funcionales, las cuales tienen el objetivo de asegurar que el sistema realice las funciones para las cuales fue definido. Las Pruebas Funcionales, al estar basadas en el análisis de los valores de entrada y resultados arrojados, utilizan los casos de prueba como herramienta de soporte para su realización.

Mediante el uso de la técnica de partición de equivalencia asociada al método de prueba de Caja Negra, los casos de prueba son divididos en secciones o escenarios según los CU del sistema, detallando las funcionalidades y variables que intervienen en los mismos. Por cada escenario de los casos de pruebas se definen un conjunto de datos de entrada (validos e inválidos) y salida para confirmar que los resultados esperados ocurran cuando se usen datos válidos y se desplieguen los mensajes apropiados de error en caso contrario.

3.2.1 Aplicación de las pruebas

Para la aplicación de las pruebas definidas, se crean varios casos de pruebas; uno por cada CU del sistema y sus diferentes secciones corresponden a los escenarios en los casos de pruebas. En el caso de prueba para el CU “Captar Formulario” se identifica la interacción de 4 variables, las cuáles se describen a continuación:

Tabla 6. Variables Caso de Prueba: CU Captar Formulario

No.	Nombre del Campo	Clasificación	Valor Nulo	Descripción
1	Acumulado hasta	JKalendar	No	Fecha del acumulado hasta ese momento, dicha fecha debe tener un formato día/mes/año.

				(Campo obligatorio)
2	Variante	ComboBox	No	Variante de la plantilla a captar. Solo números pertenecientes al conjunto de los naturales. (Campo obligatorio)
3	Unidad de Observación	Campo de texto	No	Número de la unidad de observación a captar, debe tener mínimo 5 dígitos máximo 15. (Campo obligatorio)
4	Observaciones	Campo de texto	Si	Comentarios sobre la Unidad de observación a captar. (Campo no obligatorio)

En la siguiente tabla se usaran los identificadores: Entrada Valida (V), Entrada Incorrecta (I) y Entrada No Admisible (N/A) para representar los diferentes tipos de datos en cada escenario del caso de prueba. En el caso de prueba se evidencian los diferentes resultados en correspondencia con los valores introducidos.

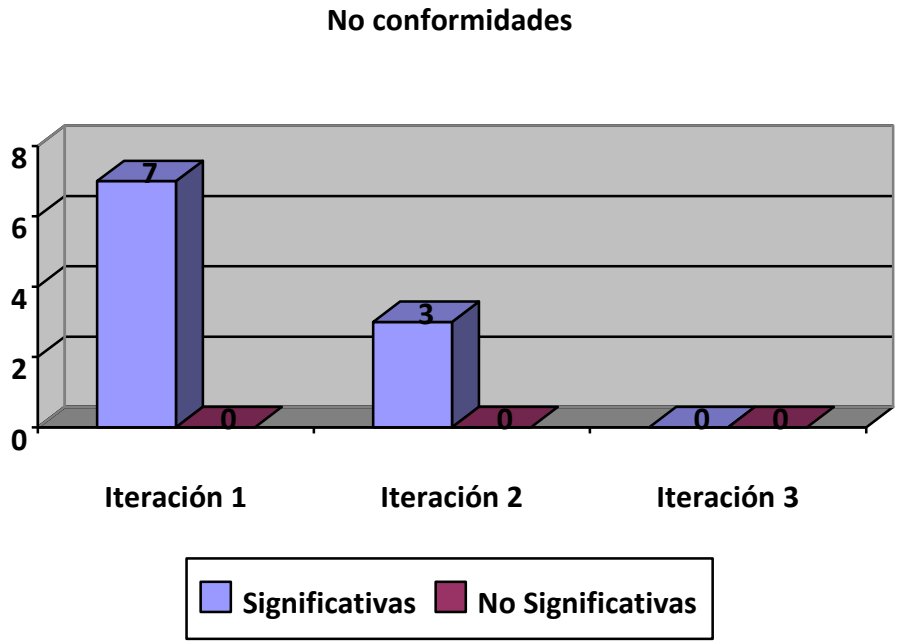
Escenario	**Descrip-ción**	**Acumulado hasta**	**Variante**	**Unidad de Observación**	**Observaciones**	**Respuesta del sistema**	**Flujo central**

EC1. Introducir correctamente los valores para realizar la captación.	Se llenan los campos correctamente y se guarda un formulario.	V (12/05/2015)	V (1)	V (910802)	V (Esto es una prueba, uno,2,3,cuatro)	Cuadro de información diciendo ¨Formulario guardado satisfactoriamente¨.	1. El Usuario introduce los datos. 2. El Usuario presiona el botón Guardar.
EC2. Dejar valores en blanco para realizar la captación.	Se desea guardar un formulario dejando campos en blanco.	V (12/05/2015)	N/A	I (1212)	V (Esto es una prueba, uno,2,3,cuatro)	Cuadro de Error diciendo ¨El código de variante seleccionado no es válido¨.	1. El Usuario introduce los datos. 2. El Usuario presiona el botón Guardar.
		V (12/05/2015)	V (1)	N/A	N/A	Cuadro de Error diciendo ¨Debe insertar una unidad de observación válida, Número de entre 5 y 15	

						dígitos¨.	
EC3. Introducir valores incorrectos para realizar la captación.	Se desea guardar un formulario con valores incorrectos.	V (12/05/2015)	I (00000)	V (123455)	V (Esto es una prueba, uno,2,3,cuatro)	Cuadro de Error diciendo ¨El código de variante seleccionado no es válido¨.	1. El Usuario introduce los datos. 2. El Usuario presiona el botón Guardar.
		V (12/05/2015)	V (2)	I (111)	V (Esto es una prueba, uno,2,3,cuatro)	Cuadro de Error diciendo ¨Debe insertar una unidad de observación válida, Número de entre 5 y 15 dígitos¨.	

Se realizaron 3 iteraciones aplicando los casos de pruebas en el sistema, en la primera iteración se encontraron 7 no conformidades (NC), en la segunda iteración 3 NC para un total de 10 no conformidades (NC), de las cuales todas se clasifican en significativas y fueron agrupadas por el equipo de desarrollo en dos tipos: de validación y de código.

Las NC fueron corregidas por el equipo desarrollador en la tercera iteración para encontrar 0 NC en las funcionalidades del sistema.



Prueba de Aceptación

Prueba de aceptación del usuario es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede usarse por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido (Pressman, 2011). (Ver anexo#2 Carta de Aceptación).

Las pruebas de aceptación pueden afrontarse mediante dos tipos de procedimiento para realizarlas: **pruebas alfa** y **pruebas beta.**

Prueba alfa: es aquella en la que se le entrega a un usuario final todo el producto terminado, junto a su documentación correspondiente para que éste, en presencia del desarrollador y en entornos previamente preparados para el proceso de dichas pruebas, vaya informando de las inconsistencias y errores que detecte (José Ponce GONZÁLEZ, 2014).

Prueba beta: es la que se le proporciona a usuarios finales, situados en lugares concretos de los puestos de trabajo donde finalmente será el software implantado, para que sea otra vez el usuario y sin contar con la presencia del equipo de desarrollo, el que de nuevo vuelva a emitir unos informes de resultados e impresiones de la aplicación o sistema software desarrollado (José Ponce GONZÁLEZ, 2014).

Luego de analizar los tipos de prueba de aceptación, se decide realizarlas de tipo alfa, la entrega a un cliente final de todo el producto terminado junto a su documentación en presencia del desarrollador, permite que las inconsistencias y errores que se detecten sean corregidos con mayor rapidez con el objetivo de satisfacer al cliente en el tiempo establecido.

Conclusiones del capítulo

Como resultado de este capítulo se obtuvo el diagrama de componentes a través de la estructura que presenta los diagramas de clases del diseño. De esta forma se realizó la implementación del sistema en términos de componentes, ofreciendo solución a los requisitos especificados en el capítulo anterior. Luego de ser generado el código fuente del sistema se aplicó la técnica de caja negra mediante la herramienta casos de prueba, lo que facilitó determinar las 10 no conformidades al culminar la primera y la segunda iteración de desarrollo. En la tercera iteración de pruebas se alcanzaron resultados satisfactorios debido a que no fue identificada ninguna no conformidad. A partir de la realización de la prueba de aceptación se validó que los requisitos funcionales definidos para el componente sean totalmente operativos elevando la calidad del software.

Conclusiones Generales

El cumplimiento de las actividades propuestas durante la investigación hizo posible que los objetivos trazados se lograran con éxito. Lo que conllevó arribar a las siguientes conclusiones:

- El análisis de SIGELITE posibilitó extraer características comunes para desarrollar la solución, así como erradicar problemas existentes e incluir las funcionalidades asociadas al módulo de encuetas para una mejor adaptación a las necesidades del cliente.

- La valoración de tecnologías, herramientas y metodología permitió realizar una adecuada elección de las mismas quedando seleccionadas: OpenUp como metodología de desarrollo; UML v2.0 como lenguaje de modelado; Visual Paradigm para UML v8.0 como herramienta CASE; Java como lenguaje de programación; NetBeans v7.3.1 como IDE de desarrollo; SQLite como Sistema Gestor de Base de Datos y SQLiteman como administrador gráfico de base de datos. Logrando que dicha selección al ser herramientas libres contribuyan con la soberanía tecnológica que pretende alcanzar el país.

- La generación de artefactos como: diagrama de CUS, diagrama de clases del diseño, diagrama de clases persistentes, diagrama de componentes, entre otros diagramas y la descripción de los CUS permitirán que otros desarrolladores obtengan un mejor entendimiento de la estructura del sistema, facilitando la realización de posibles modificaciones o la adición de nuevas funcionalidades al mismo.

- Se implementó la herramienta SIGELITE DESKTOP cumpliendo con las funcionalidades definidas por el cliente en los requisitos funcionales.

- Se comprobó el correcto funcionamiento de los 61 requisitos funcionales de la herramienta desarrollada mediante la realización de las pruebas de software.

Recomendaciones

Después de haber cumplido con los objetivos trazados en el presente trabajo de diploma se proponen la siguiente recomendación:

- Incorporar un módulo de validación para los formularios del sistema.
- Incorporar la funcionalidad que permita interpretar las fórmulas de los indicadores descritas en el diseño de la plantilla.
- Incorporar módulo de seguridad o autenticación de usuario en el sistema.

Referencias Bibliográficas

Oracle Corporation. Oracle y Java. *Oracle y Java.* [En línea] [Citado el: 25 de febrero de 2015.] http://www.oracle.com/lad/technologies/java/overview/index.html.

Apache Friends. XAMPP Apache + MySQL + PHP + Perl. [En línea] [Citado el: 22 de 2 de 2015.] https://www.apachefriends.org.

Autores, Varios. Java. *Java. ¿Qué es la tecnología Java y por qué lo necesito?* [En línea] [Citado el: 4 de diciembre de 2014.] http://www.java.com/es/download/faq/whatis_java.xml.

—. 2000. Programacion en castellano. *Programacion en castellano.* [En línea] 2000. [Citado el: 23 de marzo de 2015.] http://www.http://programacion.net/articulo/catalogo_de_patrones_de_diseno_j2ee__y_ii:_capas_de_negocio_y_de_integracion_243/8.

—. 2009. Tecnología y Synergix. [En línea] 2009. [Citado el: 25 de enero de 2015.] http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/.

BWT SIME. 2001. La Revista del empresario cubano. [En línea] 2001. [Citado el: 5 de 6 de 2015.] http://www.betsime.disaic.cu/secciones/eco_enemar_07.htm.

Canós, José H, Letelier, Patricio y Penadés, Carmen. 2008. *Metodologías Ágiles en el Desarrollo. .* Valencia: s.n., 2008.

Centro de Investigaciones para la Industria Minero-Metalúgica. 2005. CIPIMM. [En línea] 2005. [Citado el: 3 de 6 de 2015.] ISSN 1992-4194.

CITMATEL. 2002. RODAS XXI: Sistema Integral Económico Administrativo. [En línea] 2002. [Citado el: 5 de 6 de 2015.] http://www.rodasxxi.cu/.

Díaz Anton, Maria Gabriela. 2003. IV Congreso. *IV Congreso.* [En línea] 2003. [Citado el: 25 de enero de 2015.] academia-interactiva.com.

Dossier. 2014. Administración gráfica de SQLITE con SQLite Manager. [En línea] 2014. [Citado el: 15 de 1 de 2015.] ile:///C:/Users/Aylin/Desktop/Hecho/Investigar/Administración gráfica de SQLITE con SQLite Manager _ Dibosa's Blog.htm.

Eclipse. 2009. Eclipse.org. [En línea] 2009. [Citado el: 12 de 12 de 2014.] http://epf.eclipse.org/wikis/openup/..

EcuRed. Flujo de pruebas de un software. *Flujo de pruebas de un software.* [En línea] [Citado el: 4 de 5 de 2015.] www.ecured.cu/index.php/Flujo_de_pruebas_de_un_software#cite_ref-1.

—. 2012. SQLite-EcuRed. *SQLite-EcuRed.* [En línea] 2012. [Citado el: 20 de 11 de 2014.] http://www.ecured.cu/index.php/SQLite.

Gamma, E., Helm R., Johnson, R y Vlissides. 1995. Design Patterns: Elements of Reusable Object-Oriented Software. [En línea] 1995. [Citado el: 6 de febrero de 2015.] http://www.weibnc.com/wp-content/uploads/brkpdfs/Design-Patterns-Elements-of-Re. ISBN 0-201-63361-2.

GESPRO. Paquete para la gestión de proyectos. **Autores, Piñero P.Y. y Colectivo de. 2013.** No. 1. pp. 45-53. , Cuba : s.n., 2013, Vol. Vol 9. ISSN 1682-2455..

Hernández Sampieri, Roberto, Fernández Collado, Carlos y Baptista Lucio, Pilar. 1998. Metodologuía de la Investigación. *Metodologuía de la Investigación.* 1998, págs. 40-60.

Jacobson. 2000. El Proceso Unificado de Desarrollo de Software. [En línea] 2000. [Citado el: 5 de febrero de 2015.] [En línea] http://en.scientificcommons.org/6960494. ISBN 84-7829-036-2..

José Ponce GONZÁLEZ, Francisco José DOMINGUEZ MAYO, Javier Jesús GUTIÉRREZ RODRÍGUEZ, Maria José ESCALONA CUARESMA. 2014. *Pruebas de aceptación orientadas al usuario: contexto ágil para un proyecto de gestión documental.* Sevilla : Ibersid., 2014. 41012.

Juventud Rebelde. 1999. OPCIONES: Semanario Económico y Financiero de Cuba. [En línea] 1999. [Citado el: 5 de 6 de 2015.] ww.opciones.cu/cuba/2013-08-02/programa-informatico/. ISSN 1563-8340 .

2010. Lenguajes de Programación. *Lenguajes de Programación.* [En línea] 2010. [Citado el: 4 de diciembre de 2014.] http://www.lenguajes-de-programacion.com/programacion-web.shtml..

Maldonado, Daniel Martin. 2008. SQLite, el motor de base de datos ágil y robusto. [En línea] 2008. [Citado el: 20 de 1 de 2015.] http://www.aplicacionesempresariales.com/sqlite-el-motor-de-base-de-datos-agil-y-robusto.html..

MORA., ROBERTO CANALES. 2007. Modelado UML con Visual Paradigm. *Modelado UML con Visual Paradigm.* [En línea] 2007. [Citado el: 5 de enero de 2015.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=vparadigm..

Mozilla Foundation. Firefox. [En línea] [Citado el: 22 de 2 de 2015.] https://www.mozilla.org.

Pressman, Roger S. 2011. *Software Engineering, a practitioner's approach. (7th edición).* McGraw-Hil : s.n., 2011. 9780071267823..

Pressman, Roger. 2005.. *Ingeniería del software: el enfoque de un profesional . .* s.l. : 6ta edición, 2005.

Rumbaugh, James, Jacobson, Ivar and Booch, Grady. *El Lenguaje Unificado de Modelado.*

SIGELITE: GESTIÓN ESTADÍSTICA EN ENTORNOS DESCONECTADOS O DE BAJAS PRESTACIONES. **Ing. Héctor Luis Reyes Zaldívar 1*, Ing. Frank González Fernández 1, Lic. Elena Leonila Fernández García2, Ing. Claudia García Suárez del Villar1, Ing. Virgilio Suárez Bello1, Ing. Alejandro González Sánchez1, Ing. Adrian Rosales Crúz1, Ing. Clara Elena B. 2014.** 2014, UCIENCIA, págs. 1-6.

Sistema de Captura de Información para la Toma de Decisiones e Inteligencia de Negocio. **Claudia Nuñez Sanz 1*, Frank González Fernández 1, Héctor Luis Reyes Zaldívar 1, Armando Robert Lobo Pérez 1, Diana Monné Roque 1, Omar Ahmed García Pérez 1, Yanet Rosales Morales 1, Nara Lidia Pérez Solá 1. 2013.** 2013, Gestión de la Información, págs. 45-51.

Sistema de Captura de Información para la Toma de Decisiones e Inteligencia de Negocio. **Nuñez Sanz, Claudia. 2013.** 4, La Habana : Ediciones Futuro, 2013, Vol. 6. 2306-2495.

Skrum Manager. Pruebas de Integracion. *Pruebas de Integracion.* [En línea] [Citado el: 12 de abril de 2015.] http://www.scrummanager.net/bok/index.php?title=Pruebas_de_integraci%C3%B3n..

Somerville. *Somerville parte III.*

SQLite. 2011. SQLite. [En línea] 2011. [Citado el: 20 de 1 de 2015.] http://www.sqlite.org/docs.html.

The Apache Software Foundation. Apach Portable Runtime Project. [En línea] [Citado el: 22 de 2 de 2015.] apr.apache.org.

TIM BOUDREAU. 2002. NetBeans: the definitive guide. [En línea] 2002. [Citado el: 13 de 12 de 2014.] books.google.com..

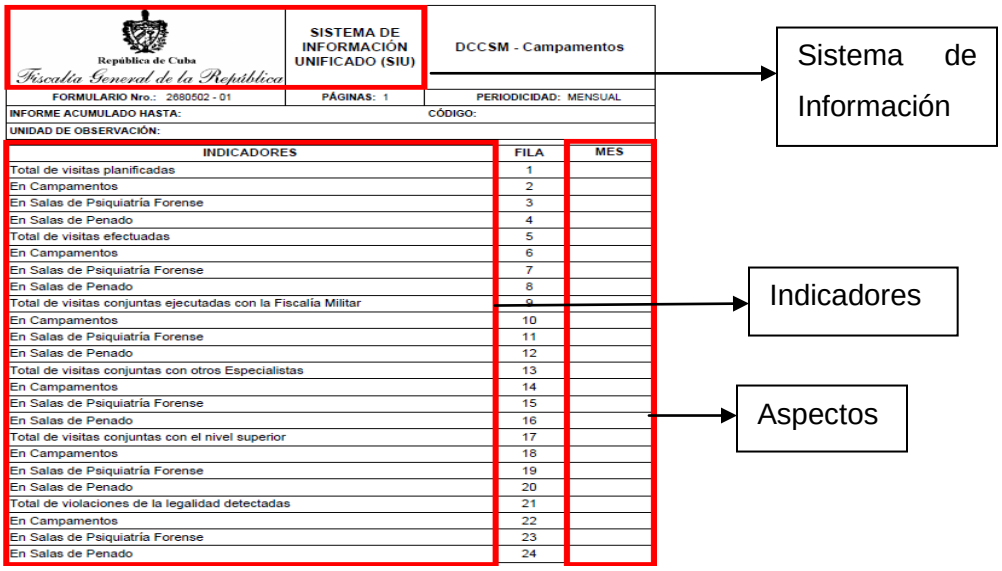
Timo Haberkern. Server2Go. [Online] [Cited: 2 22, 2015.] www.server2go-web.de.

UN MODELO DE PERSISTENCIA PARA EL DISEÑO DE LA BASE DE DATOS BASADO EN UN MODELO DE OBJETOS FORMAL. **González, Anaisa Hernández. 2002.** No. 3, 2002, Vol. Vol. 23. IO-23302-3.

Viñolo, Raydel Raúl and Roquero Figueroa, Alexander. 2012.. *Sistema Gestor de Procesos de Media v2. .* 2012.

Anexos

Anexo #1 Ejemplo de un formulario



República de Cuba
Fiscalía General de la República

SISTEMA DE INFORMACIÓN UNIFICADO (SIU)	DCCSM - Campamentos
FORMULARIO Nro.: 2680502 - 01	PÁGINAS: 1	PERIODICIDAD: MENSUAL
INFORME ACUMULADO HASTA:	CÓDIGO:
UNIDAD DE OBSERVACIÓN:	

INDICADORES	FILA	MES
Total de visitas planificadas	1	
En Campamentos	2	
En Salas de Psiquiatría Forense	3	
En Salas de Penado	4	
Total de visitas efectuadas	5	
En Campamentos	6	
En Salas de Psiquiatría Forense	7	
En Salas de Penado	8	
Total de visitas conjuntas ejecutadas con la Fiscalía Militar	9	
En Campamentos	10	
En Salas de Psiquiatría Forense	11	
En Salas de Penado	12	
Total de visitas conjuntas con otros Especialistas	13	
En Campamentos	14	
En Salas de Psiquiatría Forense	15	
En Salas de Penado	16	
Total de visitas conjuntas con el nivel superior	17	
En Campamentos	18	
En Salas de Psiquiatría Forense	19	
En Salas de Penado	20	
Total de violaciones de la legalidad detectadas	21	
En Campamentos	22	
En Salas de Psiquiatría Forense	23	
En Salas de Penado	24	

Anexo #2 Carta de Aceptación

Centro de Tecnologías de Gestión de Datos

ACTA DE ACEPTACIÓN

Por medio del presente documento se hace constar que el Trabajo de Diploma fue probado y cumple con cada uno de los requisitos definidos durante la fase de Análisis y Diseño. De igual manera el producto en sentido general satisface las necesidades del cliente.

Entrega: Tesistas

Nombre y apellidos:

Aylin Lugo Camacho

Alejandro Gracia Hernández

Tesis: SIGE Lite Desktop

Firma:

Recibe: Proyecto SIGE

Nombre y apellidos

Ing. Alejandro González Sánchez

Cargo: Líder del proyecto SIGE

Firma:

Representante Parte Suministradora

Nombre y Apellidos: Ing. Glennis Tamayo Morales

Cargo: Jefa de Departamento Componentes Informáticos

Firma:

Fecha: 29/05/2015

Anexo #3 Guía de entrevista

Objetivo: Entender el negocio.

Entrevistado: Lic. Elena Gonzales Morales.

Ocupación: Especialista de la ONEI.

Preguntas para la entrevista:

1. ¿Por qué surge SIGELITE, si ya existía SIGE?
2. ¿Cuáles son las principales deficiencias que presenta el uso de SIGELITE en las unidades de observación?
3. ¿Cuáles son las limitaciones tecnológicas que poseen las unidades de observación?
4. ¿Qué tiene que tener una nueva versión de SIGELITE en comparación con la existente y por qué?

Anexo #4 Guía de entrevista

Objetivo: Obtener los requisitos funcionales y no funcionales de SIGELITE.

Entrevistado: Ing. Héctor Luis Reyes Zaldívar.

Ocupación: Líder del proyecto de desarrollo SIGE.

Preguntas para la entrevista:

1. ¿Cuáles son las principales deficiencias que existen en SIGELITE a la hora de captar datos?
2. ¿Cuáles son las principales funcionalidades que deben implementarse en SIGELITE DESKTOP?
3. ¿Cuáles son los requerimientos de software que necesita SIGELITE tener instalados para llevar a cabo el proceso de captura de datos con éxito?
4. ¿Cuáles son los requerimientos de hardware que necesita SIGELITE tener instalados para llevar a cabo el proceso de captura de datos con éxito?