

Tipo de artículo: Artículo original
Temática: Soluciones informáticas
Recibido: 20/01/17 | Aceptado: 15/02/17 | Publicado: 27/02/17

Sistema de Gestión de Incidencias del Grupo Azucarero AZCUBA

Incident System Management of AZCUBA Sugar Group

Tania Elena Castro Aguilera^{1*}, **Bernardo Hernández González**², **Dionis Pérez Pérez**³, **Bárbara Sepúlveda mateo**⁴

¹ Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños, km 2 ½, Boyeros, Ciudad de La Habana, Cuba. taniaec@estudiantes.uci.cu

² Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños, km 2 ½, Boyeros, Ciudad de La Habana, Cuba. bhernandez@uci.cu

³ Grupo AZCUBA. Carretera a la CUJAE Km 1½, Boyeros, La Habana, Cuba. dionis.perez@azcuba.cu

⁴ Grupo AZCUBA. Carretera a la CUJAE Km 1½, Boyeros, La Habana, Cuba. barbara.sepulveda@azcuba.cu

* Autor para correspondencia: taniaec@estudiantes.uci.cu

Resumen

Con el creciente desarrollo de las tecnologías de la información y las comunicaciones, diversos procesos que se llevan a cabo en las organizaciones se han favorecido sustancialmente, al punto de ser la automatización sinónimo de eficiencia. En la actualidad, el control de incidencias en el Grupo Azucarero AZCUBA, se realiza de forma manual. Esto genera demora en la gestión, pérdida de información y falta de claridad en el análisis de los informes, influyendo negativamente en las producciones azucarera y agroindustrial a nivel nacional. En el siguiente trabajo se describe un sistema informático que permitirá la gestión de incidencias del Grupo Azucarero AZCUBA. El sistema será desarrollado con tecnologías libres: Javascript ES 6 como lenguaje de desarrollo, SailJS y AngularJSv1.5.4 como marcos de trabajo, NodeJS v7 como entorno de ejecución y PostgreSQL v9.3 como gestor de bases de datos. Como herramientas serán utilizadas: Visual Paradigm 8.0 para la construcción de los artefactos generados por la metodología y NetBeans v8.0 como entorno integrado de desarrollo. La metodología OpenUP será utilizada para guiar el proceso de desarrollo. Además, se hacen un conjunto de pruebas al sistema para validar su correcto funcionamiento.

Palabras clave: control de incidencias, Grupo Azucarero AZCUBA, sistema informático de gestión

Abstract

With the growing development of the technologies of information and communication, various processes carried out in organizations have been favored substantially, to the point of being automation synonymous with efficiency. Nowadays, the control of incidents in the Sugar Group AZCUBA, is done manually. This leads to delays in management, loss of information and lack of clarity in the analysis of reports, negatively influencing sugar and agroindustrial production at national level. The following work describes a software that will allow incident management for the Sugar Group AZCUBA. The software will be developed with free technologies: Javascript ES 6 as development language, SailJS and AngularJS v1.5.4 as frameworks, NodeJS v7 as runtime environment and PostgreSQL v9.3 as database manager. Visual Paradigm 8.0 for the construction of the artifacts generated by the methodology and NetBeans v8.0 as an integrated development environment, will be used. The OpenUP methodology will guide the development process. In addition, a set of tests are done to the system to validate its correct functioning.

Keywords: control of incidents, Sugar Group AZCUBA, management software

Introducción

Con el triunfo de la revolución cubana en 1959, se produjeron grandes cambios en la economía del país, y una de las industrias que garantizó el comercio de Cuba con el mundo fue la industria azucarera, la cual es de las más importantes de la producción agroindustrial cubana de todos los tiempos, además de ser una de las principales fuentes de ingreso a la economía del país. En el año 2011 a raíz del proceso de perfeccionamiento empresarial se disuelve el Ministerio del Azúcar y se crea la Organización Superior de Dirección para la Agroindustria Azucarera en forma abreviada AZCUBA, la cual se crea con los bienes y recursos del extinto Ministerio del Azúcar, además de los que integran su patrimonio como son sus empresas agroindustriales, productivas, de servicios, comerciales y de proyecto, contando también con los centros de Investigación de la Caña de Azúcar (INICA), el centro de Investigación de los Derivados de la Caña de Azúcar (ICIDCA) y el Centro de Capacitación (CNCA). El Grupo Azucarero AZCUBA forma parte de la administración central del estado, por su importancia económica y productiva. Su misión es la producción de azúcar y derivados de la caña, energía eléctrica y alimento animal, con calidad y costos competitivos, aplicando la ciencia y la técnica y protegiendo el medio ambiente (AZCUBA, 2014).

Las Tecnologías de la Información y las Comunicaciones (TICs) están cada vez más presentes en el mundo moderno con implicaciones en cada una de las ramas de la sociedad y el sector azucarero no está exento de este cambio, con lo que se hace más fácil en muchas de las tantas áreas que tiene este sector el trabajo a realizar, al estar las empresas sumidas en estos cambios tecnológicos pues se tiene que priorizar su correcto estado utilizable, por lo que un mal funcionamiento o la interrupción de estos servicios pueden llegar a tener importantes consecuencias en la obtención de los objetivos de la empresa.

En el Grupo Azucarero AZCUBA, entre las distintas direcciones que tiene se encuentra la Dirección de Informática, Comunicaciones y Análisis, constituida por el conjunto de salas de control y análisis existentes en el grupo, sus empresas y Unidades Empresariales de Base (UEB) y constituye el centro de dirección, información y análisis que mediante la información operativa y estadística a tiempo real, controla las 24 horas del día el flujo de los procesos, la ejecución de la zafra azucarera y las labores agropecuarias, los hechos extraordinarios y las situaciones de excepción, llamándolas incidencias en general, generando los análisis que faciliten la toma de decisiones a la dirección, y rectorando la actividad de informática, comunicaciones y archivo (AZCUBA, 2012).

En estos momentos para llevar a cabo la recepción de las incidencias se utiliza una plantilla en formato Word que la envían las direcciones que trabajan directamente con el grupo, detallando la información de lo ocurrido, con otras informaciones como quien lo reporta, fecha, hora, cargo del encargado de reportar el hecho, que no siempre es quien lo informa, y un número consecutivo, para poder darle seguimiento. Luego esa información se transcribe hacia el sistema actual creado en Visual FoxPro lo que provoca una brecha al dar la posibilidad de introducir errores al copiar y pegar, además este sistema utilizado por los directivos del nivel central solamente permite, registrar la información y generar reportes sencillos, no siendo la solución más eficiente para esta problemática porque no permite que en las unidades subordinadas se puedan crear los reportes mediante el mismo sistema, hay que hacerlo manualmente, por lo que ocurren errores humanos, como falta de información o pérdida de la misma, inconsistencia, falta de claridad en los datos, por lo que se hace difícil el correcto manejo de este tipo de información en algunos casos de gran importancia para el país, ya sea en costo monetario o de recursos materiales, ejemplo de esto la quema de la caña de azúcar, influyendo negativamente en los análisis que podrían realizar los especialistas para tomar alguna decisión.

Todo lo anterior influye de manera negativa en la toma de decisiones y la calidad y disponibilidad de la información que se gestiona. Partiendo de esta problemática se propone como objetivo de la investigación desarrollar una aplicación informática para gestionar las incidencias del Grupo Azucarero AZCUBA.

Materiales y métodos

Se propone realizar un sistema informático que permita gestionar el proceso de incidencias que se lleva a cabo en el Grupo Azucarero AZCUBA. Este sistema permitirá tener mejor control de las incidencias, mediante el cual los especialistas de cada área de trabajo involucrado podrán realizar análisis de datos específicos y necesarios por nivel organizativo lo que incidirá directamente en un mejor proceso de control y organización.

Diseño del sistema

Un actor es una entidad externa al sistema representada por un ser humano, una máquina o un software que interactúa con el sistema. Representa un tipo particular de usuario del negocio más que un usuario físico, debido a que varios usuarios físicos pueden realizar el mismo papel en relación al negocio (Schmuller, 2010). En la Tabla 1 se muestran los actores del sistema y las funciones correspondientes a cada uno:

Tabla 1. Actores del Sistema

Actor	Descripción
Usuario	Es el usuario básico del sistema, con acceso a la autenticación y con un mínimo nivel de privilegios.
Analista	Es el usuario del sistema, con permisos para gestionar las incidencias y los reportes relacionados con las incidencias.
Especialista superior	Es el usuario encargado de listar las incidencias para darle seguimiento en el sistema y además puede obtener reportes del comportamiento de las mismas, también puede exportar a los formatos Excel y PDF.
Administrador del sistema	Es el súper usuario del sistema, con permisos para agregar, eliminar y asignar privilegios a los usuarios.

Los Diagramas de Casos de Uso del Sistema (DCUS) describen parte del modelo de casos de uso y muestran un conjunto de casos de uso y actores con una asociación entre cada par de estos que interactúan en el sistema (Jacobson *et al.*, 2000b). A esto se puede agregar que constituye una excelente representación del contexto del sistema, ya que

sirve como herramienta de comunicación para visualizar, especificar, resumir y documentar el comportamiento de este y sus actores. A continuación se muestra mediante la Figura 1 el DCUS de la solución propuesta.

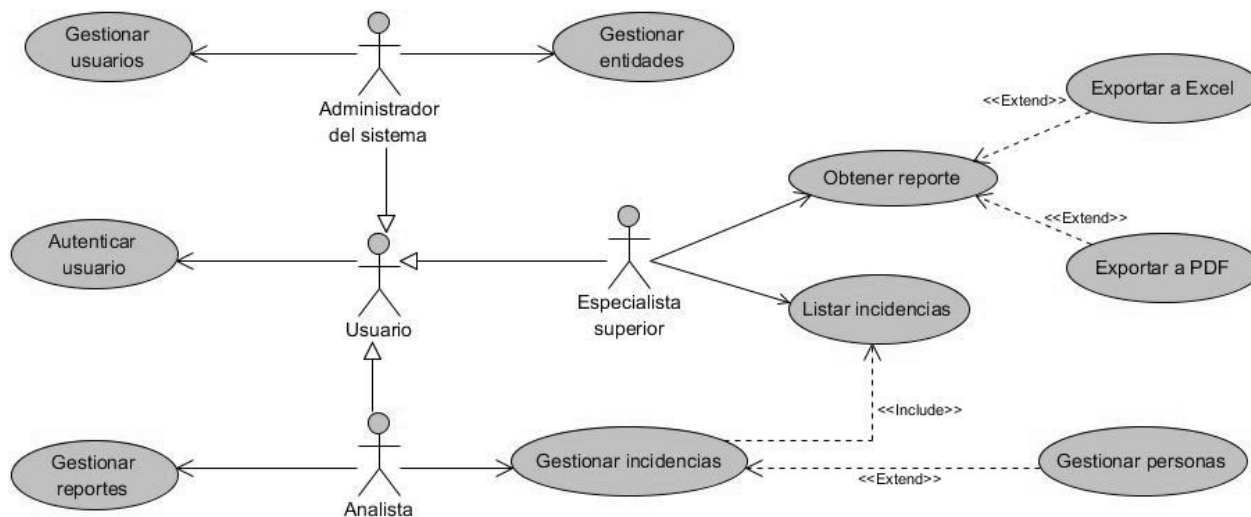


Figura 1. Diagrama de Casos de Uso del Sistema

Herramientas y tecnologías

Una vez analizadas las tendencias en el desarrollo de software se definieron las herramientas y tecnologías necesarias para la implementación del sistema, las cuales impulsan la soberanía tecnológica propuesta por el país (Mar and Jiménez, 2009). Se decidió la utilización de la metodología de desarrollo OpenUp, como lenguaje de modelado UML 2.0, como herramienta CASE Visual Paradigm en su versión 8.0, como *frameworks* de desarrollo Sail Js y Angular Js v 1.5.4, utilizando el lenguaje de programación Javascript ES6, como entorno de ejecución Node JS V7 y, como entorno de desarrollo NetBeans en su versión 8.0, PostgreSQL 9.3 como SGBD, así como la herramienta PgAdmin III 1.18 para su administración.

Arquitectura del sistema

Según Sommerville, el diseño arquitectónico o arquitectura del software es la primera etapa en el proceso de diseño y representa un enlace crítico entre los procesos de ingeniería de diseño y de requerimientos. Está relacionado con el

establecimiento de un marco estructural básico que identifica los principales componentes de un sistema y las comunicaciones entre estos componentes (Sommerville, 2005).

Un patrón arquitectónico brinda la descripción de un problema particular y recurrente de diseño, que aparece en contextos de diseño específico, y presenta un esquema genérico demostrado con éxito para su solución (Buschmann *et al.*, 2007).

Para el desarrollo de la propuesta de solución se define el patrón arquitectónico MVC, el cual se basa en las ideas de reutilización de código y la separación de conceptos, buscando facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento (Mar *et al.*, 2014). Su principal característica radica en que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos, el modelo, la vista y el controlador, los cuales se detallan a continuación:

El modelo: es la capa donde se trabaja con los datos, por tanto, contendrá mecanismos para acceder a la información y también para actualizar su estado. En la solución propuesta esta capa agrupa los paquetes “Entity” y “Model”. El primero incluye dos paquetes: “Entity”, que recoge las entidades del sistema que se crean a partir de las tablas de la base de datos, y “Repository” que contiene entidades que se encargan del manejo y selección de datos. Por su parte, en el paquete “Model” se incluyen las clases del dominio y la lógica de datos.

Las vistas: se encargan de presentar la información del sistema al usuario y generar los eventos de la interacción con éste. Capturan eventos del usuario y se los envía al sistema a través del controlador. Posteriormente reciben una respuesta del controlador y muestra información al usuario. En la propuesta de solución esta capa contiene los paquetes “View”, “Angular” y “CSS”. El primero recoge las páginas y plantillas que conforman al sistema. El paquete “Angular” contiene los archivos “Javascript” y el “CSS” las hojas de estilo.

El controlador: contiene el código necesario para responder a las acciones que se solicitan en la aplicación, sirve de enlace entre las vistas y el modelo, respondiendo a los mecanismos que puedan requerirse para implementar las necesidades de la aplicación. En la solución propuesta esta capa contiene los controladores del sistema, quienes se encargan de crear y devolver una respuesta, una vez que han recibido una petición. Para esto se utilizan clases que son destinadas a la lógica de control y otras correspondientes a los formularios, las cuales se agrupa el paquete “Controller”.

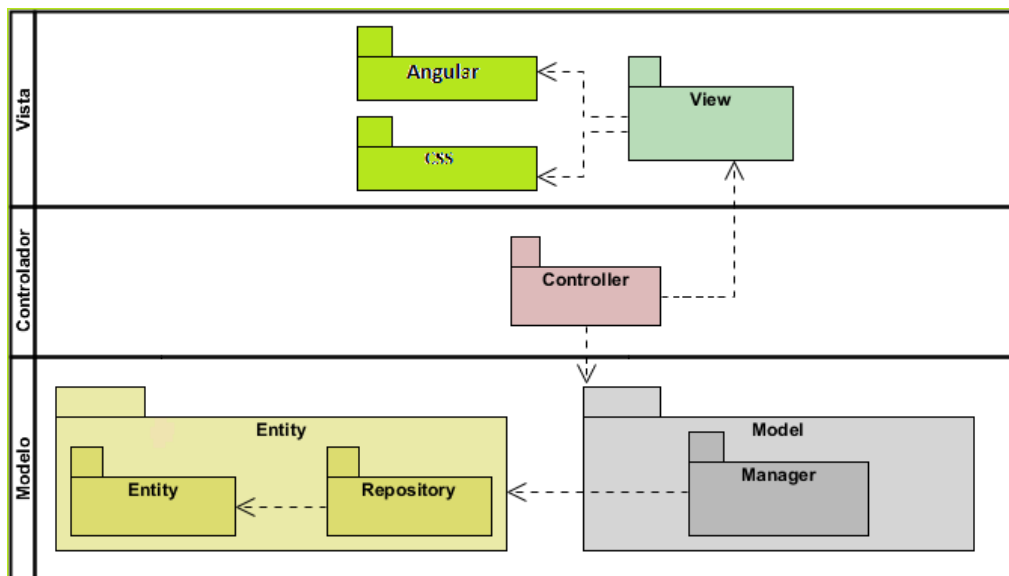


Figura 2. Vista global de paquetes de diseño del Caso de Uso: Gestionar Incidencia

Modelo de datos

Un modelo de datos es un conjunto de conceptos que sirven para describir la estructura de una base de datos, es decir, los datos, las relaciones entre los datos y las restricciones que deben cumplirse sobre los datos (Marqués, 2011). Un modelo de datos es un lenguaje orientado a hablar de una base de datos. Típicamente un modelo de datos permite describir:

- Las estructuras de datos de la base: El tipo de los datos que hay en la base y la forma en que se relacionan.
- Las restricciones de integridad: Un conjunto de condiciones que deben cumplir los datos para reflejar la realidad deseada.
- Operaciones de manipulación de los datos: típicamente, operaciones de agregado, borrado, modificación y recuperación de los datos de la base (Rumbaugh et al., 2000).

A continuación se presenta el modelo de datos de la presente investigación.

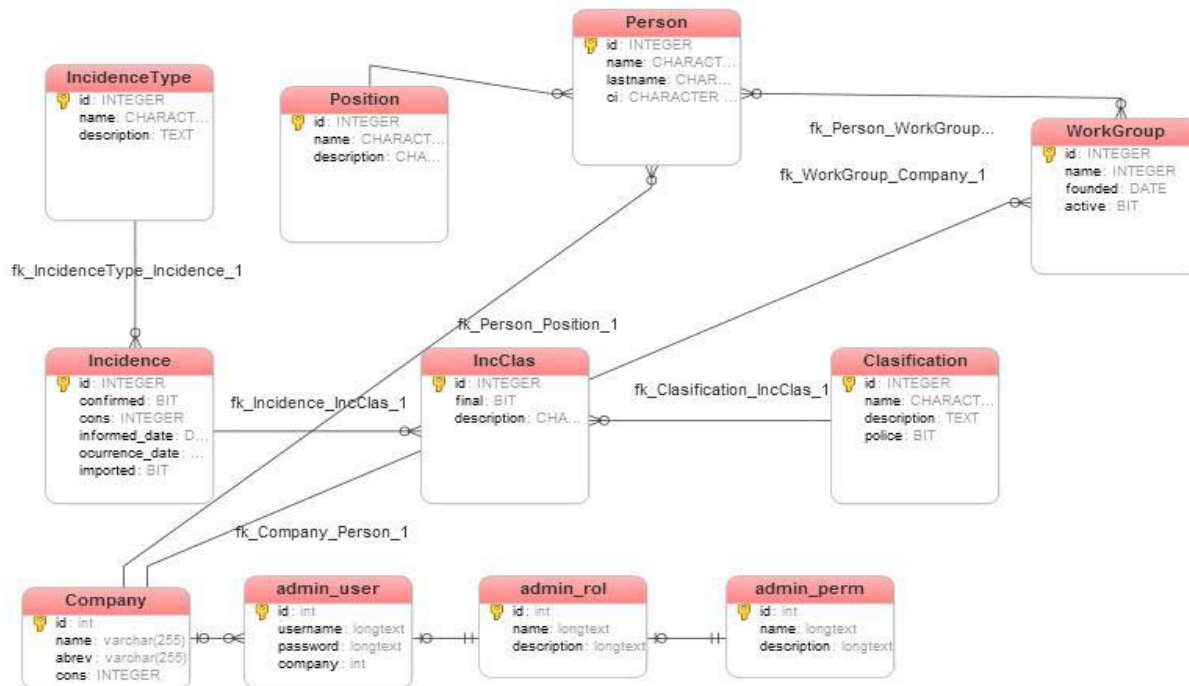


Figura 3. Modelo de datos.

Modelo de Despliegue

El propósito del modelo de despliegue es capturar la configuración de los elementos de procesamiento y las conexiones entre estos elementos en el sistema. Permite el mapeo de procesos dentro de los nodos, asegurando la distribución del comportamiento a través de aquellos que son representados (Pressman, 2010). En la Figura 4 se presenta el diagrama de despliegue propuesto para el sistema y en la Tabla 2 la descripción de sus nodos.



Figura 4. Diagrama de despliegue del sistema

Tabla 2. Descripción de los nodos correspondientes al diagrama de despliegue del sistema.

Nodos	Descripción
PC_Cliente	Nodo que representa aquellas PC que pueden ser utilizadas por los usuarios para acceder a la aplicación.
Servidor Web	Nodo que representa el servidor donde está alojada la aplicación destinada a la gestión de incidencias del Grupo Azucarero AZCUBA.
Servidor de Base de Datos	Nodo que representa el servidor donde se encuentra la Base de Datos que contiene los datos persistentes de la aplicación.

Resultados y discusión

Pruebas del sistema

Existen varios tipos de pruebas, algunas recomendadas específicamente para aplicaciones web, como las que se utilizan en esta investigación y que a continuación se explican:

Pruebas de función o a nivel de componentes, que ejercitan el contenido y las unidades funcionales dentro de la aplicación y se enfocan sobre un conjunto de pruebas que intentan descubrir errores en la misma.

Pruebas de desempeño o de rendimiento, que se aplican para descubrir problemas debido a la falta de recursos en el lado del servidor, ancho de banda de red inapropiado, capacidades inadecuadas de bases de datos, defectuosas o débiles capacidades del sistema operativo, funcionalidades mal diseñadas y otros conflictos de hardware o software que pueden conducir a un pobre desempeño cliente - servidor (Pressman, 2008).

Pruebas de aceptación, que representan aquella fase del ciclo de vida de desarrollo de software en el que el equipo de desarrollo y el área usuaria de un sistema de información tienen que garantizar que el sistema desarrollado se corresponde con los requerimientos definidos (González et al., 2014).

En la presente investigación se utilizaron estas pruebas específicamente las de tipo Alfa, que son las que realiza el usuario final, una vez recibido el producto terminado y su documentación, de conjunto con los desarrolladores del sistema. Para ello se le entregó al especialista superior el software terminado, junto a una guía para el desarrollo de estas pruebas. Este proceso se realizó en la Dirección de Informática, Comunicaciones y Análisis del Grupo

Azucarero AZCUBA, en presencia de los desarrolladores y los especialistas del área, que se encargaron de comprobar todas las funcionalidades y de informar de las inconsistencias y errores que detectaron.

Para desarrollar las pruebas de función o a nivel de componentes se aplica el método de prueba de Caja Negra, también denominado Pruebas de Comportamiento. Este método permite obtener un conjunto de condiciones de entrada que ejerciten por completo los requisitos funcionales de un programa, ignorando la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

Para desarrollar las pruebas de desempeño o de rendimiento se pueden aplicar pruebas de carga y de tensión. En el caso de la presente investigación debido a que la propuesta de solución durante su despliegue no estará expuesta a grandes niveles de tensión ni concurrencia significativa de usuarios, se define realizar solamente pruebas de carga, que tienen como objetivo determinar cómo la aplicación en su ambiente del lado del servidor responderá ante varias condiciones de carga, a partir de condiciones de pruebas definidas por las permutaciones entre variables que dependen del número de usuarios concurrentes, el número de transacciones en línea por usuario por unidad de tiempo y la carga de datos procesada por el servidor por transacción (Pressman, 2008).

Como parte de la ejecución de las pruebas de Caja Negra, con el objetivo esencial de identificar en qué medida satisface la aplicación las funcionalidades implementadas, se realizó una primera iteración de pruebas, en la que fueron aplicados los diseños de casos de pruebas realizados a partir de las descripciones textuales de los Casos de Uso del Sistema. En esta primera iteración se identificaron un total de 23 No Conformidades, distribuidas conforme se muestra en la figura 4. Una vez corregidas las No Conformidades anteriores, se ejecutan nuevamente los diseños elaborados en una segunda iteración obteniéndose un total de ocho No Conformidades. Luego de corregidas estas últimas No Conformidades se decide la realización de una tercera iteración de los diseños anteriormente aplicados obteniéndose resultados satisfactorios por lo que se determina la no realización de una nueva iteración.

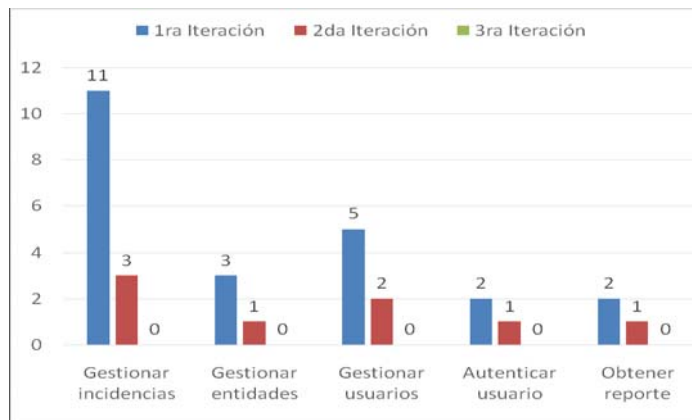


Figura 4. Resultado de la aplicación de las pruebas.

Al realizarse la ejecución de las pruebas de carga, se obtuvieron los resultados obtenidos a partir del uso de la herramienta Apache JMeter 2.3.4. Los resultados obtenidos como parte de la aplicación de la prueba de carga, realizada con una simulación de 50 usuarios concurrentes en el sistema. En la figura 5 se reflejan datos como el tiempo de respuesta en milisegundos y los errores mediante la columna Status para cada petición y la media del tiempo de respuesta, que no sobrepasa los 5 segundos conforme a lo definido en el Requisito No Funcional de eficiencia.

Muestra #	Start Time	Thread Name	Label	Tiempo de Muestra (ms)	Status	Bytes
20	22:44:43.297	Grupo de Hilos 1-20	/ceo/listar_solicitudes	108	▲	12556
21	22:44:43.378	Grupo de Hilos 1-21	/ceo/listar_solicitudes	100	▲	12556
22	22:44:43.473	Grupo de Hilos 1-22	/ceo/listar_solicitudes	109	▲	12556
23	22:44:43.555	Grupo de Hilos 1-23	/ceo/listar_solicitudes	104	▲	12556
24	22:44:43.636	Grupo de Hilos 1-24	/ceo/listar_solicitudes	108	▲	12556
25	22:44:43.722	Grupo de Hilos 1-25	/ceo/listar_solicitudes	113	▲	12556
26	22:44:43.812	Grupo de Hilos 1-26	/ceo/listar_solicitudes	121	▲	12556
27	22:44:43.909	Grupo de Hilos 1-27	/ceo/listar_solicitudes	118	▲	12556
28	22:44:43.986	Grupo de Hilos 1-28	/ceo/listar_solicitudes	105	▲	12556
29	22:44:44.082	Grupo de Hilos 1-29	/ceo/listar_solicitudes	123	▲	12556
30	22:44:44.164	Grupo de Hilos 1-30	/ceo/listar_solicitudes	115	▲	12556
31	22:44:44.246	Grupo de Hilos 1-31	/ceo/listar_solicitudes	118	▲	12556
32	22:44:44.327	Grupo de Hilos 1-32	/ceo/listar_solicitudes	136	▲	12556
33	22:44:44.408	Grupo de Hilos 1-33	/ceo/listar_solicitudes	107	▲	12556
34	22:44:44.488	Grupo de Hilos 1-34	/ceo/listar_solicitudes	107	▲	12556
35	22:44:44.576	Grupo de Hilos 1-35	/ceo/listar_solicitudes	111	▲	12556
36	22:44:44.657	Grupo de Hilos 1-36	/ceo/listar_solicitudes	121	▲	12556
37	22:44:44.738	Grupo de Hilos 1-37	/ceo/listar_solicitudes	119	▲	12556
38	22:44:44.818	Grupo de Hilos 1-38	/ceo/listar_solicitudes	115	▲	12556
39	22:44:44.901	Grupo de Hilos 1-39	/ceo/listar_solicitudes	102	▲	12556
40	22:44:44.981	Grupo de Hilos 1-40	/ceo/listar_solicitudes	126	▲	12556
41	22:44:45.065	Grupo de Hilos 1-41	/ceo/listar_solicitudes	152	▲	12556
42	22:44:45.146	Grupo de Hilos 1-42	/ceo/listar_solicitudes	106	▲	12556
43	22:44:45.226	Grupo de Hilos 1-43	/ceo/listar_solicitudes	117	▲	12556
44	22:44:45.307	Grupo de Hilos 1-44	/ceo/listar_solicitudes	112	▲	12556
45	22:44:45.387	Grupo de Hilos 1-45	/ceo/listar_solicitudes	101	▲	12556
46	22:44:45.473	Grupo de Hilos 1-46	/ceo/listar_solicitudes	113	▲	12556
47	22:44:45.549	Grupo de Hilos 1-47	/ceo/listar_solicitudes	120	▲	12556
48	22:44:45.645	Grupo de Hilos 1-48	/ceo/listar_solicitudes	120	▲	12556
49	22:44:45.735	Grupo de Hilos 1-49	/ceo/listar_solicitudes	119	▲	12556
50	22:44:45.815	Grupo de Hilos 1-50	/ceo/listar_solicitudes	106	▲	12556

Nombre: Ver Resultados en Árbol
 Comentarios
 Escribir todos los datos a Archivo
 Nombre de archivo Navegar... Log/Display Only: Escribir en Log Sólo Errores Successes

No. de Muestras 50 Última Muestra 106 Media 121 Desviación 21

Figura 5. Interfaz de salida de JMeter.

Conclusiones

El análisis de los principales conceptos asociados a la gestión de incidencias y de las principales soluciones existentes, logró que se identificaran las tendencias y las características de estas que pudieran aportar al diseño de la solución.

La realización del análisis y diseño utilizando la metodología Open UP contribuyó a obtener los artefactos para el desarrollo del sistema permitiendo su implementación.

La implementación del Sistema de Gestión de Incidencias para el Grupo Azucarero AZCUBA permitió que se obtuviera una aplicación que mejorara el control de los procesos que se desarrollan en la entidad.

La validación de la solución mediante la aplicación de las pruebas funcionales, de rendimiento y aceptación permitió obtener un sistema funcional que respondiera a las necesidades del cliente.

Referencias

- ABRAN, A.; J. W. MOORE, et al. Guide to the software engineering body of knowledge: 2004 version SWEBOK. IEEE Computer Society, 2004. p. 0769523307
- AZCUBA, G. A. Portal Web de la industria azucarera cubana, 2014. [Disponible en: <http://www.azcuba.cu/>. Reglamento del Grupo Azucarero AZCUBA. 2012. 359 p.
- BUSCHMANN, F.; K. HENNEY, et al. Pattern-oriented Software Architecture: on patterns and pattern language. John wiley & sons, 2007. p. 8126512830
- DAVIS, G. B. and M. H. OLSON. Management information systems: conceptual foundations, structure, and development. McGraw-Hill, Inc., 1984. p. 0070158282
- ESTUPIÑÁN DÍAZ, S. and I. VARGAS VARGAS Mainpack 10.0. Software para la gestión de la actividad de mantenimiento en la industria azucarera ICIDCA. Sobre los Derivados de la Caña de Azúcar, 2015, 49(2).
- FOWLER, M. Patterns of enterprise application architecture. Addison-Wesley Longman Publishing Co., Inc., 2002.
- GONZÁLEZ, J. P.; F. J. DOMINGUEZ, et al. Pruebas de aceptación orientadas al usuario: contexto ágil para un proyecto de gestión documental. España, Universidad de Sevilla, 2014.
- IEEE IEEE Recommended Practice for the Adoption of Computer-Aided Software Engineering (CASE) Tools, 1995.
- JACOBSON, I.; G. BOOCCH, et al. El proceso unificado de desarrollo de software. Madrid, España, Pearson Educacion S.A., 2000a. 464 p.
- JACOBSON, I.; G. BOOCH, et al. Rational Unified Proces, 2007.
- JACOBSON, I.; G. RUMBAUGH, et al. El proceso unificado de desarrollo de software/The unified software development process. Pearson Educación, 2000b.
- LARMAN, C. UML y Patrones. Pearson, 1999. UML y Patrones. 2da. Prentice Hall, 2003.
- MAR, O. AND JIMÉNEZ, R. Tendencias de la distribución de software, 2016.
- MAR O., GONZÁLEZ L., BRON B., DÁVILA Y., Aplicación informática para la gestión de recursos a través del directorio activo de la Universidad de las Ciencias Informáticas, 2014.
- MARQUÉS, M. Bases de datos. Castelló de la Plana: Publicacions de la Universitat Jaume I. Servei de Comunicació i Publicacions, 2011.
- PEÑA, J. M. F. Pruebas de Software, 2006. [Disponible en: <http://www.uv.mx/personal>
- PRESSMAN, R. S. Ingeniería de Software Un Enfoque Práctico. 6ta. 2005. Ingeniería de Software. Un enfoque práctico. 5ta. 2002a.
- PRESSMAN, R. S. Ingeniería del Software. 5ta. La Habana, 2002b. 579 p.
- PRESSMAN, R. S. Ingeniería del Software, 2008.

RUMBAUGH, J.; G. JACOBSON, et al. El lenguaje unificado de modelado: manual de referencia. Addison Wesley, 2000.

SOMMERVILLE, I. Ingeniería del software. Pearson Educación, 2005.