Sistema Automatizado para la Gestión Ambiental Empresarial. Módulo Monitoreo

Ing. Eduardo Javier Berrio Turiño[1], Ing. Manuel Alejandro Naranjo Rey[2], Ing. Adrián Marcos Quinta[3]

1, 2, 3. Universidad de Matanzas, eduardo.berrio@umcc.cu

Resumen

Actualmente el desarrollo de las principales actividades económicas de un país están en manos de sus empresas e indiscutiblemente la realización de las mismas traen consigo consecuencias para el medio ambiente, a pesar de los disímiles esfuerzos que realiza nuestro país para llevar a cabo un desarrollo sostenible, Cuba no está excluida de esta realidad, de ahí la necesidad de realizar un monitoreo periódico en las empresas cubanas para conocer el impacto medioambiental que están teniendo sus actividades y el estado ambiental que poseen, el problema en este proceso está en la gestión de los datos obtenidos del monitoreo, existen softwares que pueden realizar dicho proceso pero no cumplen con las normas cubanas establecidas por el Ministerio de Ciencia, Tecnología y Medio Ambiente. El módulo "Monitoreo" de este sistema permite conocer el estado actual de la calidad ambiental de una empresa.

Palabras claves: *Monitoreo; sostenibilidad; desarrollo; medioambiente.*

Monografías 2020

ISBN: 978-959-16-4472-5

Introducción

El deterioro acelerado y creciente del medio ambiente es hoy unos de los problemas más graves que enfrenta toda la especie humana en su conjunto. En lo que respecta a los países subdesarrollados es uno de los factores que agrava con más fuerza las condiciones de vida de cientos de millones de personas. Como dijera (Castro Ruz, 1992), hoy más que nunca urge a estos países el acceso al saber porque les permitiría la solución de múltiples problemas.

Con el rápido y progresivo avance de las tecnologías en el mundo, estas han pasado ya a formar parte de nuestra vida cotidiana, con el fin de satisfacer distintas necesidades en las diferentes esferas de la sociedad. En la actualidad la utilización y manejo de la información constituye un recurso vital, sobre todo en el mundo empresarial donde con su uso se pueden llegar a alcanzar grandes niveles de desarrollo, a pesar de que no siempre tienen en cuenta al Medio Ambiente y llegan a causarle daños, en ocasiones irreversibles, de ahí la necesidad e importancia de llevar cabo un desarrollo sostenible para reconciliar los aspectos económico, social, y ambiental de las actividades humanas; tres pilares que deben tenerse en cuenta por parte de las comunidades, tanto empresas como personas.

El estado cubano siempre ha sentido la responsabilidad de hacer suyo el cuidado del medio ambiente, hoy se puede afirmar que no todos los países les han dado la importancia que esto requiere, Cuba a pesar de ser un país subdesarrollado ha empleado disímiles recursos y esfuerzos para llevar a cabo esta actividad. A pesar de todo esto aún algunos sectores, especialmente el empresarial, enfocan su atención principalmente en alcanzar altos niveles de producción y pasan por alto el impacto que tienen sus actividades sobre medio ambiente, por ello el Ministerio de Ciencia Tecnología y Medio Ambiente (CITMA) emitió en 2002 la Resolución No.111 la cual establecía el Sistema Nacional de Monitoreo Ambiental cuyo objetivo es valorar el estado del medio ambiente para contribuir a la toma de decisiones sobre la protección ambiental y el uso sostenible de los recursos naturales, a través de la realización del monitoreo ambiental .

Actualmente el monitoreo en las empresas lo efectúa un especialista, el cual registra los valores obtenidos a partir de mediciones hechas a indicadores para posteriormente emitir un informe acerca del estado ambiental que posee el área en estudio. Todo el proceso de gestión de datos o sea el registro de las mediciones, la comparación de los valores recogidos con las normas que establece el CITMA y la asignación de medidas a indicadores, posee poca organización, ya que la información no está centralizada, y se realiza de forma manual por lo que se hace un proceso muy engorroso para las personas que intervienen en él, afecta su eficiencia, incrementando el margen de error y limita las posibilidades de análisis para la toma de decisiones.

Para estas organizaciones se ha impuesto entonces la necesidad de un monitoreo que suministre información en forma de indicadores e índices, que permitan evaluar la

dimensión ambiental del desarrollo y a manera prospectiva los escenarios futuros y los principales riesgos que se ciernen sobre él.

Desarrollo

Metodología de desarrollo.

Prodesoft (Proceso de Desarrollo y Gestión de Proyectos de Software), es una metodología que posee una gran flexibilidad y adaptabilidad, su proceso es iterativo e incremental, es decir, un enfoque en el que el ciclo de vida está compuesto por iteraciones, estas son pequeños procesos compuestos de varias actividades cuyo objetivo es integrar una parte del sistema parcialmente completo, probado, integrado y estable. (Xetid, 2012)

Es basado en componentes, lo que permite alcanzar un mayor nivel de reutilización de software, aún en contextos distintos a aquellos para los que se diseñó. Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados. Dado que un componente puede ser construido y luego mejorado continuamente, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo. (Xetid, 2012)

Su ciclo de vida consta de 5 fases:

- ✓ Inicio: En esta fase realiza un análisis de la problemática existente y se establece la estrategia a seguir para realizar la captura de requisitos, así como un estimado del alcance del proyecto.
- ✓ Modelación: Se capturan las partes esenciales del sistema donde se identifican los procesos de negocio fundamentales, se aceptan los requerimientos funcionales y se definen la arquitectura de sistema y de datos.
- ✓ Construcción: Es la fase donde se aclaran los requisitos restantes y se completa el desarrollo del sistema sobre una base estable de la arquitectura. En esta fase todas las características, componentes, y requerimientos deben ser integrados, implementados, y probados en su totalidad, obteniendo una versión liberada del producto.
- ✓ Explotación Experimental: Durante esta fase se convierte la versión liberada del producto en una solución estable, donde se eliminan los errores que surgen durante las pruebas y se obtiene una certificación funcional y de seguridad del producto.
- ✓ Despliegue: En la fase de despliegue se instala y configura el sistema para un ambiente de producción real, se capacita al personal que usará la aplicación y se continúa dando soporte durante la explotación del sistema.

Herramientas y tecnologías.

Lenguaje de modelado. UML (*Unified Modeling Language*): es el lenguaje de modelado de sistemas de *software* más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de *software*. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software. (Salinas Caro, Histchfeld K)

Los lenguajes de programación dan la capacidad al programador de especificar, qué tipo de datos actúan y que acciones tomar bajo una variada gama de circunstancias, utilizando un lenguaje relativamente próximo al lenguaje humano.

PHP: es un lenguaje de programación que se conoce como lenguaje de script o interpretado. Significa que el código que se escribe no se compila, sino que es interpretado por un núcleo o compilador en tiempo de ejecución. Es un lenguaje que nació para darle dinamismo a la Web. Su característica más notable es la posibilidad de embeberse junto al código HTML. (Lerdorf, 2014)

Java Script: es el lenguaje de programación web del lado del cliente más extendido, permite crear funcionalidades específicas, con él se pueden generar páginas dinámicamente en función de las preferencias del usuario, validar datos en un formulario o modificar dinámicamente el contenido de la página, puede ser ejecutado sin la necesidad de instalar otro programa para ser visualizado, es un lenguaje más orientado a objetos. (Flanagan, 2002)

Marcado de Hipertexto (HTML): es el lenguaje que permite la generación de hipertextos en la *World Wide Web*. Los lenguajes de marcado de hipertexto construyen un conjunto de reglas que definen todo aquello que es parte de un documento digital, pero que no pertenece al texto del mismo. Definen la estructura y la semántica de un documento. (*World Wide Web Consortium*, 2014). Hoy en día HTML 5 está tan de moda por las ventajas que ofrece que es imposible dejar de hacer uso de él por las nuevas etiquetas que incluye para video, audio y canvas. (Gouchat, 2012) (Albers, 2010)

Hojas de Estilo en Cascada (CSS): (del inglés *Cascading Style Sheets*) es un lenguaje formal usado para definir cómo se va a mostrar un documento en la pantalla, a imprimir o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. Se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. (No definido, 2008)

Lenguaje Estructurado de Consultas (SQL): brinda la posibilidad de realizar consultas con el objetivo de recuperar información de las bases de datos de manera sencilla. Es a la vez un lenguaje fácil de comprender y una herramienta completa para la administración de los datos.

Herramientas de desarrollo.

Gestor de Base de Datos: PostgreSQL. Permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación. Es un sistema relacional que facilita a los usuarios describir los datos que serán almacenados en la base de datos junto con un grupo de operaciones para manejar los mismos. Incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. Además, es multiplataforma y fácil de administrar. (Sanz C., 2012)

PhpStorm: es un entorno de desarrollo integrado que permite programar en PHP, HML y JavaSript, permite la gestión de proyectos fácilmente, proporciona un fácil autocompletado de código, soporta el trabajo con PHP5, resalta los nombres de las funciones y clases, identifica variables y encuentra posibles errores. (I.M.A., 2014)

Visual Paradigm-UML. Visual Paradigm-UML es una herramienta Computer Aided Software Engineering o Ingeniería de *Software* Asistida por Computadora (*CASE*) destinada a aumentar la productividad en el desarrollo de software reduciendo el coste de la misma en términos de tiempo y de dinero. Puede ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. (slideshare,2017)

Patrones de diseño.

El desarrollo de software es un proceso complejo, en especial si se demanda una solución rápida, adaptable y estable, lo que conlleva enfrentar retos continuamente. Con la utilización de patrones de diseño se puede evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente. El diseño del SAPGAE se rige por el uso de diferentes patrones implementados en el *Framework Zeolides*.

El *Framework Zeolides* usa un estilo arquitectónico híbrido, combina el estilo arquitectónico N-Capas y el Modelo Vista Controlador (MVC), tomando lo mejor de los

dos, divide la funcionalidad de una aplicación en tres componentes fundamentales: modelo, vista y controlador y a su vez divide la arquitectura de la aplicación en capas: presentación, negocio, acceso a datos y datos.

Otro elemento del estilo arquitectónico es la incorporación de requerimientos no funcionales a las aplicaciones sin necesidad de modificar el código de la aplicación utilizando los principios de la programación orientada a aspectos. Por último y no menos importante el uso del patrón inversión de control para separar las responsabilidades de una aplicación en componentes, integrando estos a través de contratos bien definidos sin necesidad de instanciar directamente sus clases o conocer sus implementaciones.

Patrón de Integración de Componentes: su intención es disminuir el acoplamiento entre componentes del *software*, evitando la instanciación directa de clases y facilitando la reutilización y la integración, localizando las interfaces o servicios que estos brindan a través de contratos bien definidos.

Modelo Vista Controlador (MVC): separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado, define componentes para la representación de la información, y por otro lado para la interacción del usuario. Este patrón de arquitectura de *software* se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

WebSSO (*Web Single Sign On*): la intención principal de emplear este patrón es solucionar que los usuarios de cualquier organización solo tengan que autenticarse una única vez para acceder a los diferentes servicios de la misma evitándoles memorizar varias cuentas de usuarios y contraseñas.

Fábrica (*Factory*): define una interfaz para crear objetos, pero deja que sean las subclases quienes decidan qué clases instanciar, permitiendo que una clase delegue en sus subclases la creación de objetos. Este patrón se utiliza cuando una clase no puede prever la clase de objetos que debe crear o cuando una clase quiere que sean sus subclases quienes especifiquen los objetos que ésta crea.

Front-Controller: obliga a que todas las peticiones hechas a la aplicación pasen por un Controlador. El controlador proporciona un punto de entrada único que controla y gestiona las peticiones web realizadas por los clientes. Normalmente se utiliza junto con un Dispatcher que es el responsable de redirigir el flujo de ejecución hacia el JavaScript adecuado. Este *Dispatcher* puede ser realizado por el propio controlador o estar en una clase a parte.

Patrones Grasp: nos indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo obtendremos un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento).

Factory Method: centraliza en una clase constructora la creación de objetos de un subtipo determinado, ocultando al cliente los casos para elegir el subtipo que crear. Clase con la responsabilidad de crear objetos de otras clases. No delega en subclases y sus métodos pueden ser estáticos. Puede evolucionar en un *Abstract Factory*.

Singleton: se asegurar de que una clase tiene una sola instancia y proporcionar un punto de acceso global a ella.

Fachada (Facade): provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema.

Observer: define una dependencia entre un objeto y un conjunto de ellos, de modo que los cambios en el primero se vean reflejados en los otros. Permite reutilizar sujetos y observadores por separado. Se pueden realizar implementaciones con observadores que coordinan información sobre varios sujetos.

Arquitectura por capas: tiene como objetivo principal la separación de las partes que componen el sistema es decir separa la lógica de negocios, la capa de presentación y la capa de datos. De esta forma permite crear diferentes interfaces sobre el sistema sin tener que cambiar la capa de datos o lógica.

Comando: Este patrón permite solicitar una operación a un objeto sin conocer realmente el contenido de esta operación, ni el receptor real de la misma. Para ello se encapsula la petición como un objeto, con lo que además se facilita la parametrización de los métodos.

Validador-Interceptor: garantiza que los datos provenientes de cualquier entidad externa, cumplan con un conjunto de reglas de sintaxis, tipo de dato, longitud y negocio.

Escape de las salidas: evita que, caracteres o información de control presentes en los datos, tengan un significado especial para el intérprete hacia el que van dirigidos.

Punto único acceso: define un punto único de acceso hacia la aplicación donde sea posible aplicar diferentes medidas de seguridad con el objetivo de ejercer un mejor control de las comunicaciones hacia y desde la misma.

Modelo de dominio.

El SAPGAE se desarrolla a partir de las diferentes necesidades que tienen en común las empresas cubanas para llevar a cabo su gestión ambiental, pero con la capacidad de

ajustarse y cumplir con los requisitos individuales de cada una de ellas. Es por ello que no se establece un modelo de negocio, sino que en su lugar se desarrolló un modelo de dominio que puede utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, por lo que puede ser tomado como punto de partida para dicho diseño. El modelo de dominio es utilizado por el análista como un medio para comprender el sector de negocios al cual el sistema va a servir. Este modelo permitirá mostrar de manera visual los principales conceptos que se manejan, ayudando a los usuarios, desarrolladores e interesados; a utilizar un vocabulario común para poder entender el contexto en que se desarrolla el sistema.

Diseño de la base de datos.

Uno de los pasos cruciales en la construcción de un software que maneje una base de datos, es sin duda, el diseño de la misma pues tiene como propósito asegurar que los datos persistentes sean almacenados consistente y eficientemente. En el diseño de la base de datos los modelos de datos tienen un papel significativo pues se encargan de proveer una vista de las entidades lógicas de datos y sus relaciones, si los modelos no son definidos apropiadamente, podemos tener muchos problemas al momento de ejecutar consultas a la base de datos.

Factibilidad.

Al desarrollarse un software es preciso saber si será factible o no su producción, por lo que hay que tener en cuenta los costos y beneficios que traerá consigo. Es necesario también realizar una estimación del esfuerzo, el tiempo de desarrollo y la cantidad de personas que participarán para poder determinar eficazmente si resulta beneficioso el desarrollo, aunque es importante señalar que solo se habla de una estimación. Para este trabajo se utilizó la metodología Prodesoft que desde un inicio estimó la duración de la implementación de cada uno de los requisitos, es importante destacar que esta metodología guía un proceso basado en componentes lo que permite ahorrar gran cantidad de tiempo. El tiempo que se empleará en cada requisito se estima basado en la experiencia del programador en el trabajo con el lenguaje de programación, el entorno de desarrollo, el conocimiento sobre el tema de investigación y las técnicas de programación necesarias para resolver el problema. La estimación por lo tanto es de forma empírica y desde el comienzo se conoce el tiempo de duración total estimado del proyecto.

Dentro de los beneficios que reporta el módulo Monitoreo cabe mencionar que, permite a la empresa realizar una evaluación rápida y confiable de forma automática de su situación respecto al medio ambiente. Este módulo provee a la empresa de una herramienta eficiente que elimina el trabajo manual, disminuye los costos y el tiempo de realización del estudio. La seguridad y protección de los datos se corresponda con el nivel requerido por el cliente y cumple con las expectativas especificadas en los requerimientos funcionales del cliente.

En resumen, se evidencia que el desarrollo de la solución propuesta es factible, ya que el balance costo/beneficio es equilibrado, con predominio de los beneficios respecto al costo. Además, es válido aclarar que el producto de software obtenido (SAPGAE) se comercializa por la Xetid, por lo que a largo plazo los ingresos que esta herramienta reportará superarán a los gastos de desarrollo.

Conclusiones

El estudio realizado sobre los antecedentes, el estado actual de la temática, la bibliografía y documentos relacionados con el objeto de estudio, permitió contar con los elementos necesarios para dar solución a la problemática planteada, pues los sistemas automatizados encontrados, vinculados al tema no le dan solución al problema planteado ya que no permiten la personalización exigida por el cliente. Se utilizaron las herramientas de software propuestas por XETID y se implementó el módulo Monitoreo del SAPGAE.

Se realizó la estimación del costo de implementación del sistema y el estudio de factibilidad, arrojando como resultado la factibilidad de la realización del sistema informático, para un producto de mayor calidad se le aplicaron pruebas al sistema, las que permitieron la detección de errores y la pronta corrección de los mismos.

La implementación del sistema y la aplicación de las pruebas de validación con resultados satisfactorios demostraron que el software elaborado cumple con los requerimientos especificados constatándose, mediante avales, su aporte práctico a la gestión empresarial cubana.

Referencias bibliográficas

ALBERS BRIAN, P. Pro HTML5 Programming, pp. 305, 2010.

CASTRO RUZ, F. Cumbre de las Naciones Unidas sobre el Medio Ambiente y el Desarrollo. Río de Janeiro,1992.

DESCONOCIDO. *Ventajas de usar CSS*. [en línea] [fecha de consulta: 14 de diciembre de 2017].
Disponible en: www.stardustxs.com/2008/03/05/ventajas-de-usar-css

FLANAGAN DAVID, D. JavaScript: The Definitive Guide. 2002.

GOUCHAT DIEGO, J. El gran libro de HTML5, CSS3 y Javascript. 2012.

I.M.A. *PHPStorm* [en línea] [fecha de consulta: 23 de diciembre de 2017]. Disponible: http://dazzet.co/7-tips-paramejorar-tu-desempeno-en-phpstorm-parte-1

LERDORF RASMUS, P. *PHP* [en línea] [fecha de consulta: 14 de diciembre de 2017]. Disponible:
www.php.net.

SALINAS CARO, P. y HISTCHFELD K, N. *Tutorial UML* [en línea] [fecha de consulta: 15 de enero de 2018]. Disponible:
http://users.dcc.uchile.cl/~psalinas/uml/introduccion.html

SANZ, C. *¿Que es PostgreSQL?* [en línea] [fecha de consulta: 14 de diciembre de 2017]. Disponible: www.nan-tic.com/es/2012/que-es-el-postgresql

WORLD WIDE WEB CONSORTIUM. *Borrador actual de especificaciones de HTML5*. [en línea] [fecha de consulta: 2 de mayo de 2018]. Disponible:
http://dev.w3.org/html5/spec/Overview.html

XETID. Proceso de Desarrollo y Gestión de Proyectos de Software. La Habana, Cuba. 2012.