Serie Científica de la Universidad de las Ciencias Informáticas
Vol. 10, No. 7, Mes Julio, 2017
ISSN: 2306-2495 | RNPS: 2343
Pág. 49-60
http://publicaciones.uci.cu

Tipo de artículo: Artículo original
Temática: Tecnologías de Datos
Recibido: 04/04/17 | Aceptado: 03/05/17

Evolución del Servidor Dinámico de Reportes

Dynamic Report Server Evolution

Antonio Barreto Sánchez [1]*, Juliét Armas Guerrero [2]*

[1] Universidad de las Ciencias Informáticas. Carretera San Antonio de los Baños km 2 1/2. abarreto@uci.cu
[2] Universidad de las Ciencias Informáticas. Carretera San Antonio de los Baños km 2 1/2. julie@uci.cu

[*] Autor para correspondencia: abarreto@uci.cu

Resumen

El dinamismo es una de las principales características de las Tecnologías de la Información y las Comunicaciones (TIC), elemento que causa la constante evolución e innovación de los conceptos, productos y servicios. En este contexto, la industria del software se está desarrollando a un ritmo acelerado, imponiendo desafíos a los equipos de desarrollo, deben cumplir con las crecientes demandas del mercado y las tecnologías que utilizan, teniendo que aplicar nuevas técnicas, procedimientos y desarrollar nuevas habilidades para asegurar el progreso de sus soluciones o hacer frente a la dramática caída de la obsolescencia en un corto período de tiempo. Este trabajo muestra el desarrollo de una herramienta de software, resultado de la aplicación de las mejores prácticas en ingeniería de software y codificación, ha refinado su negocio y ampliado su acción, como es el caso del Servidor Dinámico de Reportes(SDR) que actualmente se encuentra por la versión 2.0, y ahora tiene una amigable y funcional interfaz gráfica de usuario (GUI), así como otras características y mejoras que se le han incorporado como parte del proceso de mejora en base a las necesidades de los usuarios finales.

Palabras clave: reporte, servidor, interfaz gráfica de usuario, innovación.

Abstract

Dynamism is one of the main characteristics of Technologies of Information and Communications (TIC), this element causes the constant evolution and innovation of concepts, products and services. In this context the software industry is developing at an accelerated pace, imposing challenges to development teams, they must comply with the ever

increasing demands of both the market and the technologies they use, having to apply new techniques and procedures, and develop new skills to ensure the progress of their solutions or face the dramatic drop in the obsolescence in a short period of time. This paper shows the development of a software tool, which result from the application of best practices in both software engineering and coding, it has refined its business and expanded its action, this is the case of the Dynamic Reports Server (SDR), which already goes by the version 2.0, and now has a friendly and functional Graphical User Interface (GUI), as well as other features that have been incorporated as part of the improvement process based on the needs of end users.

Keywords: *report, server, graphic user interface, innovation.*

Introducción

El escenario actual de las instituciones que hacen ciencia está marcado por un ciclo de vida corto de los productos, mercados globales, competidores desconocidos, necesidades cambiantes de los clientes, entornos dinámicos, tecnología compleja, con compromisos competitivos determinado por el desarrollo de nuevos productos y que su explotación sea rápida y con una clave para la ventaja competitiva sustentada en crear competencias tecnológicas en la institución (Díaz-Balart, 2004).

En este contexto se desarrollan las Tecnologías de la Información y las Comunicaciones (TIC), que conceptualmente según lo entendido por Guiza (Guiza, 2011) consisten en el «conjunto de tecnologías que permiten la adquisición, producción, almacenamiento, tratamiento, registro y presentación de información en formatos de sonido, imágenes y datos que incluyen la electrónica como su tecnología base, que apoya el desarrollo de las telecomunicaciones, la informática y el audiovisual», pero más que resultados constituyen procesos, donde cada objetivo logrado es el punto de partida a uno o varios objetivos que se desencadenan, fomentando la innovación y transformación constante de las metas obtenidas.

De ahí que la capacidad de la empresa informática para ser competitiva en este nuevo escenario dependerá de su habilidad para incorporar en el menor tiempo posible las competencias y aptitudes directamente relacionadas con la gestión del cambio (innovación), la gestión del conocimiento y la actualización de este activo a través de una formación de excelencia (Drucker, 1993).

Esta vertiginosidad con que ocurren los cambios científicos y tecnológicos en la informática, expresión de la potencia revolucionaria de esta rama de la ciencia, pone en constante tensión a los desarrolladores de software. Por lo que los equipos de desarrollo deben trazar estrategias acertadas desde la planificación inicial y a largo plazo, donde quede

garantizado el constante ajuste a las exigencias crecientes, teniendo en cuenta la necesidad futura de aplicar nuevas técnicas y procedimientos, la ampliación de las funcionalidades de los sistemas, y la formación de nuevas habilidades, algunos de los factores que influyen en el progreso estable de las soluciones informáticas. O de lo contrario deben enfrentar la dramática caída en la obsolescencia en un relativo corto período de tiempo.

Entre otras medidas estos factores quedan mitigados con anterioridad por la definición certera de la arquitectura del software, puesto que la arquitectura del software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución (ISO/IEC/IEEE, 2017), involucrando todos los elementos que permitirán al sistema crecer de forma natural, destacando decisiones tempranas de diseño que tendrán un profundo impacto en todo el trabajo de ingeniería de software que le sigue, y determinando el éxito final del sistema como una entidad operacional.

Materiales y métodos

Fue necesario guiar la investigación utilizando métodos científicos, tanto teóricos como empíricos. Los métodos teóricos permitieron estudiar las características del objeto de la investigación. Dentro de estos se empleó, el **analítico – sintético**, para buscar la esencia de los fenómenos y caracterizar los elementos más importantes relacionados con los motores de reportes más utilizados y con mayores prestaciones. Además, ayudó en el establecimiento de los métodos, herramientas y procedimientos más factibles para la implementación del servidor de reportes que se propone.

También se empleó el método **inductivo – deductivo** para generalizar el conocimiento adquirido luego de aceptada la idea a defender y determinar que el desarrollo de un servidor para la generación de reportes es una idea bien aceptada. Otro método utilizado es el **histórico – lógico** con el objetivo de actualizar los antecedentes y el estado actual del desarrollo de motores de reportes, así como los logros y limitaciones de los mismos en Cuba y en el mundo.

La arquitectura del SDRv2.0 se basa en el estilo arquitectónico en Capas, específicamente en 2 capas aplicando el estilo de Llamada y Retorno, y se utiliza la estrategia de diseño arquitectónico basada en patrones. Entre los patrones de diseño seleccionados para garantizar el éxito de la propuesta se encuentran los patrones generales de software para asignación de responsabilidades (GRASP): Experto en información, Controlador, Alta Cohesión, Bajo Acoplamiento, Polimorfismo y Creador. Se definió el estándar de codificación para garantizar con su aplicación el uso mínimo de recursos en la implementación del código fuente, la rapidez en la lectura y comprensión del mismo, así como la facilidad en su depuración, mantenimiento y escalabilidad.

Herramientas y tecnologías

Se determinó el uso de las siguientes herramientas y tecnologías para el desarrollo exitoso de la aplicación informática, todas bajo licencia de software libre:

- Sistema operativo Ubuntu v12.04.
- Metodología de Desarrollo de Software: AUP.
- Lenguaje de Modelado: UML v2.1.
- Herramienta CASE: Visual Paradigm for UML v8.0.
- Lenguaje de Programación: Java (JDK v1.7).
- Entorno Integrado de Desarrollo (IDE): NetBeans v7.4.
- Gestor de Base de Datos: PostgreSQL v9.1.
- Herramienta de administración de la base de datos pdAdmin III v1.14.0.
- Servidor de Aplicaciones: Apache Tomcat v7.0.34.0.
- Motor de Reportes: JasperReports v5.0.4.
- Servicios de Transferencia de Estado Representacional (REST).
- Librería Quartz v1.8.4.
- Herramienta de control de versiones Subversion v1.6.17.
- Framework Qooxdoo v5.0.1.

Resultados y discusión

El Servidor Dinámico de Reportes (SDR) surgió con el fin de dar soporte al dinamismo en la generación de reportes a partir de la información almacenada en una fuente de datos, en su versión 1.0 se implementó bajo el principio de brindar una capa de servicios que permitiera su integración con cualquier otro sistema, web o de escritorio, con independencia de sus plataformas de desarrollo, a través del cual los clientes pudieran generar reportes en una gran variedad de formatos sobre múltiples fuentes u orígenes de datos. Una solución general para el almacenamiento, compilación y exportación de reportes, sin especificaciones a la medida de un cliente determinado. Además el servidor permite la gestión de suscripciones o como son nombrados en el negocio del sistema, tareas programadas,

que provee facilidades a los usuarios de planificar con una frecuencia predeterminada el envío de exportaciones de reportes específicos por correo electrónico o a través de servidores FTP[1].

Sin embargo, como su funcionamiento se basa en servicios web que son consumidos por las aplicaciones clientes sin necesidad de una interfaz de usuario que medie entre los dos sistemas, no existía una Interfaz Gráfica de Usuario (GUI por las siglas en inglés de Graphic User Interface) a través de la cual pudieran realizarse las peticiones, determinando que para el consumo de los servicios del SDR v1.0 se requería en las aplicaciones clientes el uso o implementación de cualquier cliente AJAX[2], CURL[3] o HTTPRequest[4] con independencia del lenguaje de programación.

Este hecho limitaba la experiencia de los usuarios de las aplicaciones clientes con el SDR, porque aun cuando no es necesaria la comunicación directa de los usuarios con el servidor, la persona puede solicitar la ejecución de los servicios directamente, y recibirá las respuestas satisfactorias correspondientes siempre que envíe las peticiones correctamente parametrizadas. Pero la característica peculiar del SDR v1.0 de no contar con una interfaz visual propia, complejizaba la forma de interactuar con él, así como el modo en que se visualizaban las respuestas, teniendo que hacer uso del intérprete de comandos del sistema operativo, donde utilizando la librería CURL se le realizan peticiones a los múltiples servicios y se recibe la respuesta correspondiente, funcionamiento que sucede internamente entre las aplicaciones clientes y el servidor de forma totalmente transparente para los usuarios. En consecuencia, documentos como el Manual de Usuarios estaba dirigido a usuarios con conocimientos avanzados de las Ciencias Informáticas, como, por ejemplo: ingenieros en informática y administradores de servidores.

A modo de ejemplo de la complejidad y sobretodo de lo incómodo del proceso de comunicación con el servidor, a continuación, se muestra una imagen con una sección del fichero ejecutable que ilustra una petición realizada al SDR v1.0 a través de CURL.

[1] FTP: Protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (Transmission Control Protocol).
[2] AJAX: Tecnología asíncrona a través de la cual los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página web.
[3] CURL: Herramienta utilizada en los intérpretes de comandos para transferir archivos con sintaxis URL.
[4] HTTPRequest: Instrumento para el desarrollo de aplicaciones informáticas que permite leer los valores HTTP enviados por un cliente durante una solicitud web.

Serie Científica de la Universidad de las Ciencias Informáticas
Vol. 10, No. 7, Mes Julio, 2017
ISSN: 2306-2495 | RNPS: 2343
Pág. 49-60
http://publicaciones.uci.cu

```
curl -i -X POST http://127.0.0.1:8084/sdr/services/rest/connection \
     -H 'X-SDR-API-Key: '$token \
     -H 'Accept:application/json' \
     -H 'Content-Type: application/json' \
     -d '{"name":"Connection 1","password":"",
        "url":"jdbc:postgresql://127.0.0.1:5432/sdr",
        "username":"postgre"}'
```

Fig. 1: Estructura para consumir un servicio del SDR v1.0 a través de CURL.

Y la consiguiente ejecución de la petición a través de la consola del sistema operativo Ubuntu 14.04, mostrada en la Fig 2.

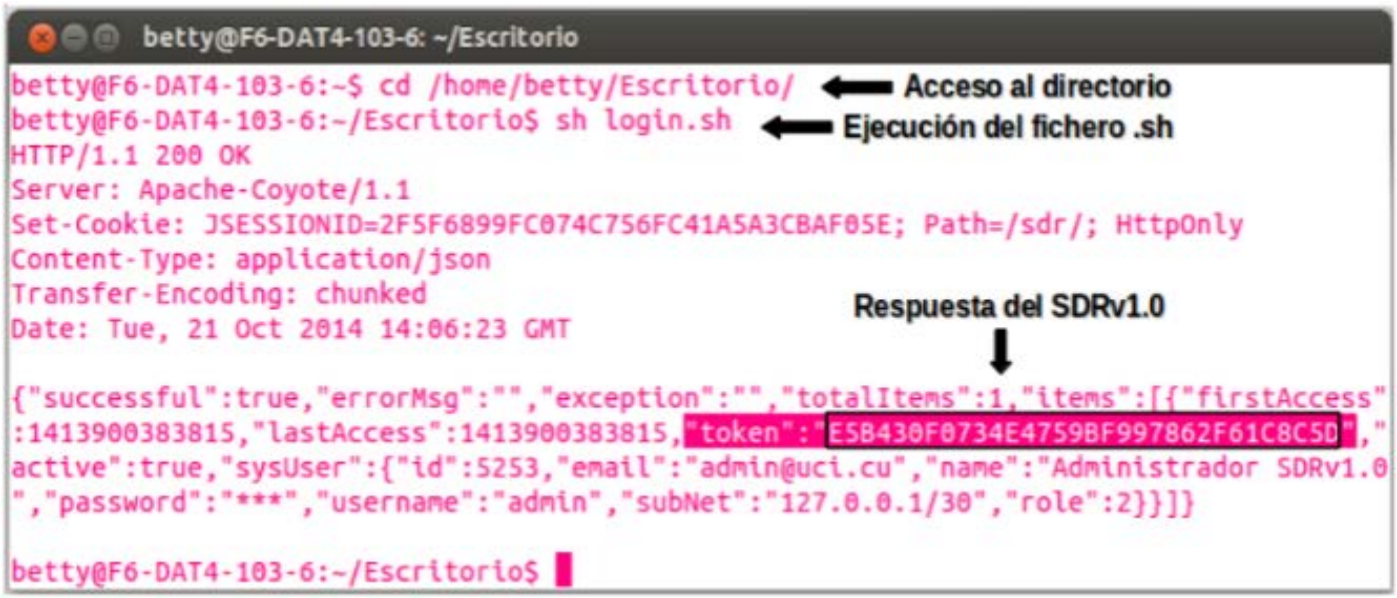

Fig. 2: Acceso y ejecución de un fichero con la petición al SDRv1.0.

Con el despliegue del SDR v1.0 a varios clientes, a pesar de la aceptación exitosa en todos los casos, el equipo de desarrollo reconoció la necesidad de implementar una aplicación web que constituyera la GUI del servidor, a través de la cual se pudiera interactuar visualmente con las funcionalidades ofrecidas por él, y que además permitiera ampliar las funcionalidades realizadas por el usuario con rol de administrador, dándole facilidades para desempeñar sus labores administrativas.

Serie Científica de la Universidad de las Ciencias Informáticas
Vol. 10, No. 7, Mes Julio, 2017
ISSN: 2306-2495 | RNPS: 2343
Pág. 49-60
http://publicaciones.uci.cu

Para el desarrollo del SDR v2.0 se siguieron las buenas prácticas aplicadas en la versión precedente desde las primeras etapas de desarrollo del proyecto, tanto en la ingeniería del software como en la codificación; empezando con la definición de la arquitectura del software, donde gracias a la flexibilidad, escalabilidad, mantenibilidad y reusabilidad en la que se soporta el diseño de la versión anterior resultó de baja complejidad la realización de los cambios y la incorporación de los nuevos elementos. Entre las múltiples definiciones que abarca la arquitectura del software, cabe destacar que se escogió continuar utilizando las tecnologías empleadas anteriormente con versiones actualizadas para la implementación de los nuevos servicios que amplían el alcance del negocio del servidor, dando más cobertura a las actividades administrativas, y la inclusión del framework de JavaScript Qooxdoo para la implementación de la interfaz visual, por medio del cual una característica nueva dentro de la arquitectura empezó a tener presencia dentro del sistema, la usabilidad, hasta el momento ignorada por la ausencia de comunicación visual. A través de este atributo de calidad quedaron garantizados importantes elementos de la aplicación web como son la comprensibilidad, la operatividad, la utilizabilidad y la conformidad, entre otras.

Esta nueva versión sigue contando con el servidor como base, con el mismo principio de funcionamiento y características de la versión 1.0, pero además provee una aplicación web creada como una solución independiente, que integrada al servidor como otro cliente garantizará el acceso gráfico a todos los servicios brindados por el SDR. Con la ampliación de las funcionalidades aumentaron los recursos que el servidor maneja, entiéndase como recurso a los conceptos gestionados por el servidor, es el caso de conexiones a fuentes de datos, categorías, reportes, exportaciones y tareas programadas. En esta nueva versión los servicios se agrupan en 7 paquetes funcionales como evidencia la Fig 3.

Fig. 3: Nuevo alcance del SDR v2.0 representado a través de paquetes funcionales.

De forma general se pueden presentar las principales características y funcionalidades del SDR v1.0 de la siguiente manera:

- Permite la generación de reportes en múltiples formatos: PDF, XML, HTML, CSV, XLS, XLSX, RTF, JPG, ODT, PPTX, DOCX, ODS y XML4SWF, a partir de diferentes fuentes de datos.
- La conexión a las fuentes de datos de los reportes se limita solo por el soporte de los API JDBC (p*or las* siglas en inglés de Java Data Base Connection) de Java.
- Permite la generación de subreportes.
- Visualiza las vistas previas de los reportes.
- Posibilita la selección de estilos de estructura para un grupo de reportes.
- No limita el tamaño del diseño del reporte para su exportación.
- Atiende peticiones de aplicaciones web o de escritorio a través de servicios REST.
- Constituye un sistema multiplataforma.
- Utiliza el gestor de base de datos PostgreSQL para persistir la información del servidor y los datos de los reportes.
- Implementar un conjunto de políticas de seguridad a varios niveles que garantiza el acceso controlado a los recursos y servicios.
- Permite la programación de tareas para la exportación automática de reportes en un momento específico, y su envío por correo electrónico o por el Protocolo de Transferencia de Archivos (*FTP por las* siglas en inglés de *File Transfer Protocol*).
- Provee funciones administrativas que garantizan la gestión de los recursos almacenados en el servidor, como la gestión de usuarios y permisos, de conexiones a las fuentes de datos, de reportes, de exportaciones de los reportes, y de tareas programadas.

El criterio esencial que rigió el desarrollo de la interfaz gráfica fue la necesidad de una aplicación muy ligera, que no constituya una recarga de trabajo para el servidor, de modo que no influya en la agilidad y rapidez del desempeño del mismo, y no afecte su rendimiento. Así se logró el objetivo planteado y en las figuras 4 y 5 se puede apreciarse una sección de la apariencia de la página principal de la aplicación web, que cuenta con dos módulos, uno para la administración del servidor, accesible solo por los usuarios que tienen asignado el rol de Administrador, que además de la gestión de usuarios, permite manipular desde el ambiente visual la configuración del servidor respecto al envío de correos electrónicos, el acceso a los ficheros que almacenan las trazas del registro de actividades realizadas en el

Serie Científica de la Universidad de las Ciencias Informáticas
Vol. 10, No. 7, Mes Julio, 2017
ISSN: 2306-2495 | RNPS: 2343
Pág. 49-60
http://publicaciones.uci.cu

servidor, incluyendo los errores ocurridos, así como la publicación y visualización de reportes de interés administrativos con el fin de optimizar el uso de los recursos de hardware; y otro módulo para el resto de los usuarios, que tienen acceso a todos los servicios vinculados con los recursos reporte, exportación, categoría, conexión a fuente de datos, y tareas programadas.

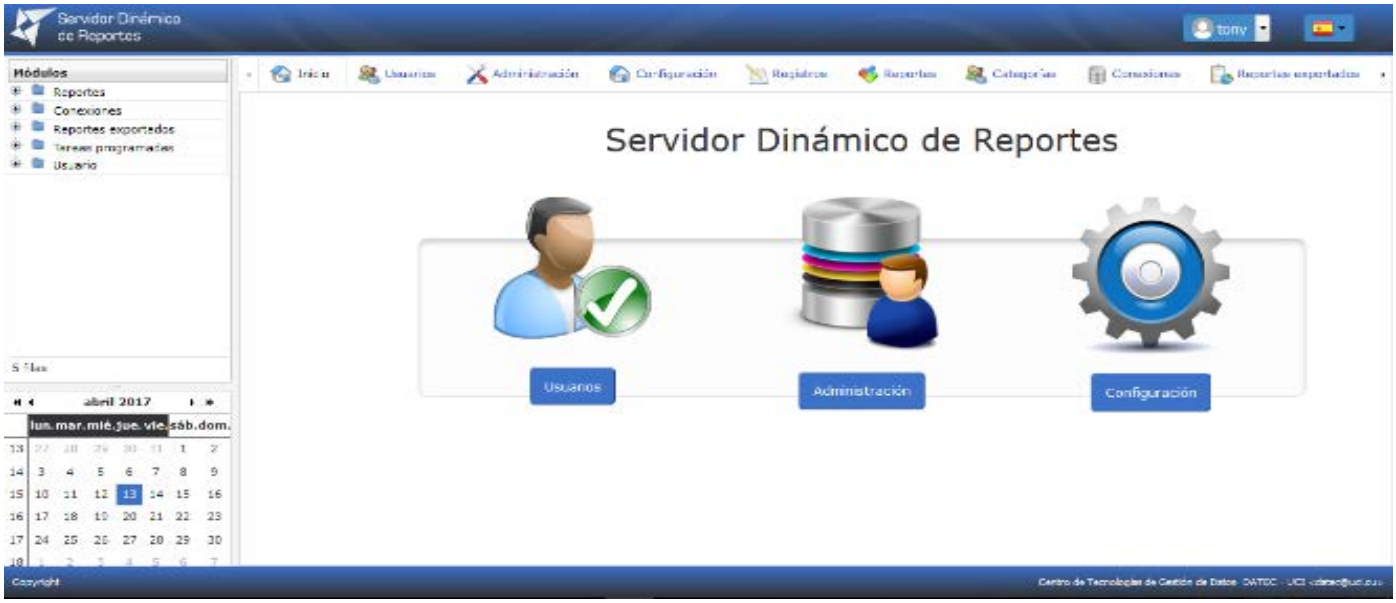

Fig. 4: Página principal del Módulo de Administración del SDR v2.0

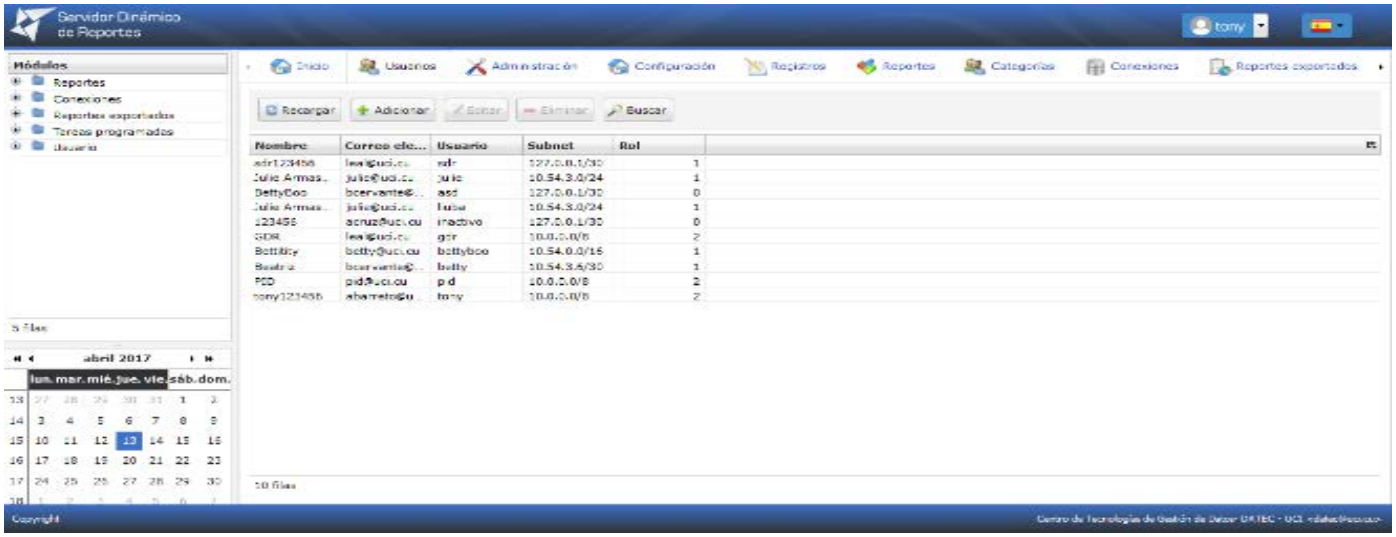

Fig 5. Página principal del Módulo de Usuarios del SDR v2.0

En cualquiera de los casos se mantienen vigentes todas las medidas de seguridad establecidas desde la versión 1.0, que garantizan el acceso controlado tanto a servicios como a recursos, y limita la acción de los usuarios solo a los recursos que él mismo ha creado.

Otro elemento del servidor que continúa en crecimiento es el listado de Sistemas de Gestión de Bases de Datos[5] (SGBD) que pueden constituir fuentes de datos para las conexiones que contienen la información mostrada en los reportes. La incorporación de nuevos SGBD depende de la solicitud de los clientes, y como se ha explicado anteriormente, se ha tenido en cuenta este escenario cambiante por lo que resulta simple y rápido el proceso de adición, hasta la redacción de este trabajo son soportados los más utilizados:

PostgreSQL, SQL Server, MySQL, SQLite, FirebirdSQL, Mondrian, MongoDB, CSV, JSON, XML, Microsoft Access, Oracle.

Pruebas de software

La prueba de software es un proceso de ejecución de un programa con la intención de descubrir errores. Las pruebas definen el grado de aceptación del sistema que pueden tener los clientes del mismo, e incluso determinar si lo aceptan o no. Para ser más efectivas, las pruebas deben ser realizadas por extraños, que puedan probarlo de forma despiadada, esto garantiza pruebas con alta probabilidad de encontrar errores. Las pruebas no pueden asegurar la ausencia de errores; sólo puede demostrar que existen defectos en el software (Pressman, 2010).

Con el objetivo de evaluar la calidad de la aplicación informática desarrollada, se diseñó una estrategia de prueba que abarcara todos los niveles: Unidad, Independiente, Desarrollador, Sistema e Integración; efectuando tanto las del tipo Caja Blanca como Caja Negra, haciendo uso de las técnicas Camino Básico y Partición Equivalente respectivamente; como instrumento se empleó el Caso de Prueba y como herramienta informática el Apache JMeter™ Desktop, herramienta que se distribuye libremente bajo la licencia de Apache.

Los resultados de las pruebas fueron satisfactorios, en la primera iteración se detectaron 12 No Conformidades de mediana complejidad que fueron resueltas y en la segunda iteración no se detectó ninguna No Conformidad, permitiendo arribar a la conclusión de que la nueva versión está en óptimas condiciones para su explotación en un ambiente de despliegue real.

Teniendo en cuenta que un elemento crucial en la implementación de la interfaz gráfica de usuario es que el funcionamiento de esta aplicación web no puede representar una afectación al rendimiento del servidor, la ejecución de pruebas de rendimiento a la solución desarrollada para todos los posibles escenarios en que pudiera estar

[5] SGBD: Conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos.

desplegado el SDR v2.0 en un entorno real constituyó un hito dentro de la realización del proyecto. La prueba de rendimiento (Carga y Stress) está diseñada para probar el rendimiento del software en tiempo de ejecución dentro del contexto de un sistema integrado y se realiza durante todos los pasos del proceso de las pruebas. Incluso al nivel de unidad, se debe asegurar el rendimiento de los módulos individuales a medida que se llevan a cabo las pruebas de Caja Blanca. Sin embargo, hasta que no están completamente integrados todos los elementos del sistema no se puede asegurar realmente el rendimiento del sistema (Pressman, 2010).

Las Pruebas de Carga se realizan para determinar y validar la respuesta de la aplicación cuando es sometida a una carga esperada en el ambiente de despliegue real. Al realizar las pruebas se comprobó la respuesta satisfactoria del servidor mediante la GUI ante la petición de exportación de 100 reportes en formato PDF en forma simultánea, con tiempos de respuestas promedios acordes a los obtenidos cuando se realizaron las pruebas en la versión anterior, concluyendo que el SDR v2.0 se encuentra preparado para dar soporte a la carga de múltiples peticiones simultáneas con bajo costo para el sistema expresado en los bajos tiempos de respuesta del servidor.

Las Pruebas de Estrés se emplean para medir la respuesta del sistema informático ante distintos volúmenes de carga esperados, en este caso cantidad de reportes exportados. Las pruebas se ejecutaron para verificar la velocidad de respuesta al procesar la exportación de 25, 50 y 100 reportes en forma simultánea a través de la interfaz visual. A partir de los valores alcanzados en los resultados del rendimiento con 25, 50 y 100 muestras de reportes a exportarse en formato PDF de forma simultánea, es evidente el buen rendimiento del sistema informático desarrollado.

Valoración económica y aporte social

La aplicación de la solución constituye un gran impacto económico, que impulsa el desarrollo local y además permite el ahorro de recursos por concepto de pago de licencias a software propietario.

A continuación, se muestran los elementos que fundamentan el impacto que tendrá la solución una vez aplicada:

- Alto nivel de gestión, control y seguridad de la información.
- Facilidad en la obtención de los reportes necesarios para dar solución a las necesidades de los clientes.
- Rapidez y calidad en la exportación de los reportes diseñados.
- Mayor número de formatos de exportación de reportes, que incluyen los formatos más utilizados mundialmente.
- Contribuye notablemente a la toma de decisiones de las instituciones, teniendo en cuenta la tendencia creciente a la acumulación de información vital almacenada en diversas fuentes de datos que requiere de gran esfuerzo y tiempo para su presentación y análisis, donde la generación de reportes surgió como el apoyo

decisivo en la realización exitosa y ágil de esos procesos; y por tanto constituye un aporte significativo para la sociedad actual y futura.

Conclusiones

- La capacidad de la empresa para ser competitiva en el actual escenario está directamente influenciada por su decisión y habilidad de desarrollar sistemas escalables y flexibles e incorporar las competencias y aptitudes necesarias para gestionar eficazmente el cambio.
- El Servidor Dinámico de Reportes es un producto que ha evolucionado satisfactoriamente, en consonancia con las exigencias de la empresa informática actual, como consecuencia de la planeación estratégica desde etapas tempranas del desarrollo del software garantizando su vigencia y continuidad.
- Las pruebas de rendimiento realizadas a la versión 2.0 permiten manifestar que la Interfaz Gráfica de Usuarios implementada para el Servidor Dinámico de Reportes no influye en la capacidad de respuesta del servidor.

Referencias bibliográficas

Díaz-Balart, F. C. (2002). *Ciencia, innovación y futuro.* Barcelona, España: Grupo Editorial Random House Mondadori.

Díaz-Balart, F. C. (2004). *Ciencia, Tecnología y Sociedad: Hacia un desarrollo sostenible en la Era de la Globalización.* La Habana, Cuba: Editorial Científico-Técnica.

Drucker, P. (1993). *Administración para el futuro.* España: Editorial Parramón.

Guiza, M. (2011). *Trabajo colaborativo en la web: Entorno Virtual de Autogestión para docentes. Tesis doctoral*. Universidad de las Islas Baleares.

Martínez, R. (2013). PostgreSQL-es. *PostgreSQL-es*. Obtenido de http://www.postgresql.org.es/sobre_postgresql

PgAdmin. (2016). *"PgAdmin"*. Obtenido de https://www.pgadmin.org/index.php

Pressman, R. S. (2007). *Ingenierá de Software, un enfoque práctico.* McGraw-Hill Companies.

Pressman, R. S. (2010). *Ingeniería de Software. Un enfoque práctico.* 7th ed.

Sommerville, I. (2006). *Software Engineering*.

Website, I. 4. (2017). *Systems and software engineering — Architecture description*. Obtenido de http://www.iso-architecture.org/42010/