Propuesta de arquitectura basada en componetes para plataformas de gestión de procesos desarrolladas en Django.

Autores:

Ing. Elena Martínez Pérez

Ing. Sergio Andres Pérez Cancio.

Resumen

El **objetivo** del presente trabajo es reflejar el aumento de la escalabilidad en los sistemas de gestión que necesitan integrarse con otros sistemas mediante una plataforma. Se realizó un estudio de los distintos estilos arquitectónicos y patrones vinculados al objeto de estudio para conformar la propuesta de una arquitectura modificable, segura y funcional, que asegure el fácil mantenimiento, amplia flexibilidad y alta reusabilidad de los componentes de la plataforma, apoyándose en el uso de herramientas libres. Se hizo un análisis del funcionamiento y de las características fundamentales de las tecnologías, herramientas y lenguajes utilizados en el diseño e implementación de la arquitectura, cuyo desarrollo de software estuvo guiado por la **metodología** ágil SXP. Como **resultado**, se logró una arquitectura basada en componentes reutilizables, eficiente y funcional; capaz de integrar nuevas aplicaciones sin que se vea afectado el rendimiento de la plataforma. Finalmente fue sometida a una evaluación con el objetivo de identificar los riesgos y fortalezas de la propuesta, para mejorar la calidad del software y su aplicación.

Abstract

The aim of this paper is to reflect the increased scalability in the management systems they need integrarses with other systems via a platform. A study of different architectural styles and patterns related to the subject of the proposed study to form a modified architecture, secure and functional, ensuring easy maintenance, high flexibility and high reusability of the components of the platform, based on the using free tools. An analysis of the operation and of the fundamental characteristics of the technologies, tools and languages used in the design and implementation of the architecture, the software development was guided by the agile methodology SXP. As a result, achieved a reusable component-based architecture, efficient and functional, able to integrate new applications without affecting the performance of a platform. Finally underwent an assessment in order to identify risks and strengths of the proposal, to improve software quality and application.

Introducción

Es imprescindible sustentar el crecimiento gradual y el enriquecimiento cognoscitivo de todas las ramas de la informática, en tanto se vive en un mundo cada vez más globalizado, donde cada día desaparecen las barreras comerciales y culturales, y la calidad aparece como una necesidad, puesto que permite competir con mayores posibilidades de éxito. En busca de la calidad del software se orienta la arquitectura de software, que provee las herramientas para realizar un producto que responda a los requisitos deseados de funcionamiento, usabilidad y rentabilidad, que son algunas de las cualidades a tener en cuenta para el desarrollo de productos informáticos.

Un témino muy usado en las universidades cubanas, en particular en las de corte informático, es plataforma para la gestión de procesos, en la cual se gestionan diversos procesos que se llevan a cabo en las universidades, en las dimensiones administrativas, docentes, investigativas y extensionistas que se desarrollan en el entorno universitario . Como ejemplo más concreto la investigación se centró en la Universidad de las Ciencias Informáticas (UCI), y en particular en la Facultad Regional Granma (FRG). La plataforma existente presentaba problemas de escalabilidad, o sea, que al añadir o aumentar algún elemento, se veía afectada sus funcionalidades, reflejada a la hora de integrar una nueva aplicación o crear un nuevo componente en la plataforma, afectándose su funcionamiento y el tiempo de respuesta. Para dar solución al problema existente era necesario desarrollar una nueva arquitectura para la plataforma. Aspecto que conlleva a la necesidad de concebir un marco o ambiente de trabajo idóneo para el desarrollo de aplicaciones adicionales que puedan insertarse o vincularse como plugins a personalizaciones desplegadas de la plataforma.

Medología ágil

El desarrollo de la arquitectura estuvo guiado por la metodología SXP la cual es una metodología ágil compuesta por SCRUM y XP que ofrece una estrategia tecnológica a partir de la introducción de procedimientos ágiles para actualizar los procesos de software. Está especialmente indicada para proyectos de pequeños equipos de trabajo, de rápido cambio de requisitos o requisitos imprecisos, donde exista un alto riesgo técnico; se orienta a una entrega rápida de resultados y una alta flexibilidad.

Al usar SXP se utilizó SCRUM para la gestión de proyectos y XP para la ingeniería de software. Esta metodología respondió de forma rápida a los cambios de acuerdo con las peticiones o necesidades del cliente, permitió la organización y gestión de los procesos de forma satisfactoria.

Desarrollo

Arquitectura de software

En la actualidad crecen las exigencias y funcionalidades del software por lo cual su complejidad de desarrollo es mayor y, como consecuencia, la arquitectura de software recibe un interés superior con el fin de dar solución a esas exigencias. Existen diversas definiciones de arquitectura de software según diferentes autores; a continuación se abordan algunas de ellas:

En el libro “Ingeniería del Software. Un enfoque práctico”, del autor Roger S. Pressman, avalada autoridad internacional y doctor en Ciencias Físicas en Ingeniería de la Universidad de Connectituc, Estados Unidos, se ofrece la siguiente definición de arquitectura de software:

"La arquitectura de software de un sistema de programa o computación es la estructura de las estructuras del sistema, la cual comprende los componentes del software, las propiedades de esos componentes visibles externamente, y las relaciones entre ellos.”[1]

En esta definición se valora la arquitectura de software como la estructura fundamental de un sistema y el papel de los componentes, lo que permite que el personal vinculado con el diseño arquitectónico pueda analizar la efectividad de los requisitos fijados y reducir los riesgos que surjan durante el proceso de desarrollo del sistema.

En el libro “El Proceso Unificado de Desarrollo de Software”, de los autores Ivar Jacobson, Grady Booch y James Rumbaugh, quienes desarrollaron el Lenguaje Unificado de Modelado (UML) y tuvieron reconocimiento mundial por sus investigaciones en el campo de las ciencias de la computación, se define la arquitectura de software como:

“... la arquitectura en un sistema software se describe mediante diferentes vistas del sistema en construcción. El concepto de arquitectura software incluye los aspectos estáticos y dinámicos más significativos del sistema. Sin embargo, también se ve influida por muchos otros factores, como la plataforma en la que tiene que funcionar el software (arquitectura hardware, sistema operativo, sistema de gestión de base de datos, protocolos para las comunicaciones en red), los bloques de construcción reutilizables de que se dispone, consideraciones de implantación, sistemas heredados, y requisitos no funcionales (por ejemplo, rendimiento, fiabilidad).” [2]

En esta definición se puede apreciar que la arquitectura es fundamental dentro del ciclo de desarrollo de un software. Además, en ella no solo se definen tecnologías o herramientas a utilizar, sino que se abordan como aspecto importante los requisitos no funcionales asociados a la construcción del software.

Se puede concluir que la arquitectura de software es una vista estructural del sistema, que define o combina estilos para dar solución al problema, abarcando incluso los requisitos no funcionales; su definición es una necesidad apremiante dentro de la ingeniería de software. En ella se enfocan a grandes rasgos todos los elementos que se deben tener en cuenta al definir algún sistema.

Descripción general

La propuesta consiste en desarrollar una arquitectura basada en componentes reutilizables, permitiendo así que las aplicaciones que se integren a la plataforma hagan uso de dichos componentes. Será como un esqueleto que puede ser utilizado por cualquier sistema de gestión que en algún momento de su aplicación necesite integrar algún componente nuevo o adicionar otras aplicaciones. La arquitectura contiene una serie de componentes reutilizables de modo que puedan generar menús de servicios, recursos asociados por las aplicaciones (CSS, JS), definir contenidos de portlet y una serie de funcionalidades para dar solución a la problemática existente. La arquitectura fue implementada en el lenguaje python el cual es dinámico y orientado a objetos, permite simplicidad, su sintaxis es clara y legible y se caracteriza por la reutilización. Además se utilizó el framework o marco de trabajo Django, con el mismo se puede crear y mantener aplicaciones web de alta calidad con un mínimo esfuerzo. Provee un alto nivel de abstracción de patrones comunes en el desarrollo, atajos para tareas frecuentes de programación y convenciones claras sobre cómo solucionar problemas.

Componentes de la arquitectura

Estos componentes son reutilizables e independientes, son usados por los sistemas que se quieren integrar a la plataforma existente.

- ContentProvider representa un proveedor de contenido para la pantalla de inicio del portal, en ella se define de forma general el contenido que se genera por las aplicaciones. Está compuesto por un grupo de variables y funciones para almacenar el título del contenido, la posición del área en el que se va alojar y el contenido a generar.

- MenuAppProvider se utiliza para generar el menú de servicio de las diferentes aplicaciones presentes. Está compuesto por un grupo de variables y funciones para almacenar el título de la aplicación, la imagen de esta y un enlace hacia la misma.

- MenuProvider se utiliza para generar el menú del portal. Está compuesto por un grupo de variables y funciones para almacenar el título del ítem para el menú y los diferentes subitems asociados al ítem generado, así como los enlaces asociados a los mismos.

- ResourseProvider se utiliza para generar los diferentes recursos asociados a las aplicaciones (CSS, JS).

- PluginMount es una metaclase que es instanciada por los componenetes mencionados anteriormente y representa un punto de montaje o punto de extensión para los plugins a generar.

Principales funciones de la arquitectura

Los componentes mencionados anteriormente son la base fundamental de la arquitectura, pero también cuenta con una serie de funciones que permiten que se complemente el trabajo final de la arquitectura. Estas funciones son las que a continuación se enuncian:

- initial_content: esta función se encarga de recolectar todo el contenido generado por las distintas aplicaciones existentes y ubicarlo en las distintas áreas en dependencia del contenido, conformando así la página principal del portal.
- load_resource: se encarga de recolectar los archivos estáticos generados por las distintas aplicaciones y ubicarlo en la página principal del portal.
- load_menu: es el encargado de generar el menú principal del portal, recolectando los diferentes ítems generados por las distintas aplicaciones para conformar así el mismo.
- load_menu_app: se encarga de generar el menú de las aplicaciones o de servicio, recolectando los datos de todas las aplicaciones existentes.
- autodiscover: es el encargado de buscar dentro de todas las aplicaciones instaladas un módulo llamado portal.py y registrar todas las clases dentro del mismo.

Validación y resultados

Para evaluar la arquitectura de la plataforma libre para la gestión de procesos de la FRG, se decide utilizar el método ATAM, para lo cual se seleccionaron los atributos de calidad que se evalúan en los escenarios escogidos, teniendo en cuenta que las propiedades o requerimientos no funcionales influyen notablemente en la calidad del software.

El Método de Análisis de Acuerdos de Arquitectura (ATAM) está basado en escenarios en los cuales se evalúan atributos de calidad y también se puede observar la interacción entre ellos. Permite describir la forma en la que el sistema puede crecer, responder a cambios e integrarse con otros sistemas. Normalmente es usado cuando los atributos de calidad, rendimiento y confiabilidad son los de mayor interés.

En la evaluación de la arquitectura se probaron cada unos de los componentes, los cuales funcionan correctamente y sin afectar el rendimiento de la plataforma. Después de analizar los resultados alcanzados con la evaluación de la arquitectura se llega a la conclusión que la misma es adecuada, proporcionando seguridad, escalabilidad, mantenibilidad y flexibilidad al sistema. Se demostró que la plataforma es suficientemente flexible para afrontar los cambios tanto en la infraestructura tecnológica como en el ambiente de negocio, lo cual posibilita que pueda crecer sin problemas. Además la arquitectura, al ser desarrollada en Django, es independiente de la plataforma. Puede ser ejecutada en cualquier sistema operativo que tenga un servidor web.

Conclusiones

Es importante que todo sistema de software esté definido por una arquitectura sólida, que permita organizar su desarrollo, fomentar la reutilización y hacer crecer el sistema, aspectos teóricos y metodológicos tenidos en cuenta en el desarrollo de la investigación, lo que permitió arribar a la siguiente conclusión.

La arquitectura de software desarrollada a partir de componentes reutilizables logra una buena escalabilidad en la plataforma libre para la gestión de procesos de la Facultad Regional Granma, con un óptimo rendimiento, avalado por las pruebas realizadas a su funcionamiento durante la etapa evaluativa. Al mismo tiempo dicha arquitectura puede ser usada por sistemas desarrollados en Django que se integren a alguna plataforma. Se han podido constatar que el diseño e implementación de componentes reutilizables e independientes en una arquitectura permite elavar su escalabilidad y rendimiento.

Referencias bibliográficas

1. PRESSMAN, Roger. Ingeniería del Software. Un enfoque práctico. 6ta. ed. La Habana: Editorial Félix Varela, 2005. 599 p.
2. JACOBSON, Ivar, BOOCH, Grady, RUMBAUGH, James. El Proceso Unificado de Desarrollo de Software [en línea]. Madrid: Addison Wesley, 2000 [fecha de consulta: 11 de enero del 2012]. Disponible en: http:// eva.grm.uci.cu.
3. CARRILLO, Isaías, PÉREZ, Rodrigo, RODRÍGUEZ, Aureliano. METODOLOGÍA DE DESARROLLO DEL SOFTWARE [en línea]. 2008 [fecha de consulta: 23 de abril del 2012]. Disponible en: http://solusoftg11.googlecode.com/files/Metodologias%20de %20desarrollo.pdf.
4. GUTIÉRREZ, Javier.\textit{¿Qué es un framework web?} [en línea]. [fecha de consulta: 20 de febrero del 2012]. Disponible en: http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.
5. HOLOVATY, Adrian y KAPLAN-MOSS, Jacob. El libro de Django [en línea]. Editor técnico: Jeremy Dunck, 2007 [fecha de consulta: 24 de febrero del 2012]. Disponible en: http://portal.frgrm.grm.uci.cu/projects/sig/files.
6. ALVAREZ, Miguel. Qué es jQuery [en línea]. [fecha de consulta: 2 de marzo del 2012]. Disponible en: http://www.desarrolloweb.com/articulos/introduccion-jquery.html.
7. BRIAN DEBUIRE, Enríquez. El Lenguaje de Programación Python [en línea]. [fecha de consulta: 25 de febrero del 2012]. Disponible en: www.ecualug.org/files/Flisol%20-%20Python.pdf.
8. EGUÍLUZ, Javier. Introducción a JavaScript [en línea]. [fecha de consulta: 3 de mayo del 2012]. Disponible en: http://www.librosweb.es/javascript/.