Máster Universitario en Software Libre

Proyecto Fin de Máster

Curso Académico 2012/2013

Metodologías, Modelos de Desarrollo y ALM

Autor: Ricardo García Fernández

Tutor:

Índice general

Capítulo 1

Introducción

SidelabCode Stack es una forja de desarrollo de Software para su uso como herramienta **ALM**. Es una herramienta FLOSS (Free Libre Open Source Software) con Licencia **TBC**.

El desarrollo de este proyecto se basa en el diseño e implementación del proceso de desarrollo acorde con las metodologías ágiles a través de la forja *SidelabCode Stack*. Unificando herramientas a través de las distintas APIs basadas en la interoperabilidad, facilitando la instalación, la replicación y la recuperación de los datos mediante el uso de Software Libre para construir Software de calidad.

El uso de metodologías ágiles se encuentra intrínsecamente relacionado a lo largo del proceso de desarrollo e implementación de la forja. Por otra parte podemos afirmar que no es una herramienta intrusiva para la aplicación de las distintas metodologías ligadas a un proceso de

desarrollo.

Presentando las distintas metodologías de desarrollo de software definidas analizando sus características tanto las buenas como las malas y como éstas se aplican al diseño de la herramienta encaminado un proceso de desarrollo fluido y ágil para un desarrollador o un grupo de desarrolladores.

Un punto a tener en cuenta es la ergonomía con la que cuenta la herramienta para el día a día y las soluciones que aporta con respecto a otras herramientas que cohabitan en el mismo campo.

Todo con un mismo fin; producir software de calidad evaluable, es decir, no es necesario crear un producto perfecto sino se sabe mejorarlo y adaptarlo a nuevas necesidades. Para eso utilizamos las herramientas ALM en los proyectos ya que nos ayudan a tener una visión diaria y a posteriori con perspectiva para poder medir y evaluar las mejoras entre dos puntos de tiempo. Por lo tanto *la evolución del proyecto y la adaptabilidad a los cambios*.

1.1. Etimología

Sidelab es, un laboratorio de software. Partiendo de esta base, encontramos una definición exacta para SidelabCode [1]:

> *Sidelab es el "laboratorio de software y entornos de desarrollo integrados"(Software and Integrated Development Environments Laboratory). Es un grupo de entusiastas de la programación con interés en prácticamente todos los aspectos del desarollo, desde los lenguajes de programación y los algoritmos avanzados, hasta la ingeniería del software y la seguridad informática. Nuestros principales intereses se centran en el desarrollo software y la mejora y personalización de los entornos de desarrollo integrados (IDEs) y herramientas relacionadas.*

En el otro extremo del nombre se encuentra *Code* se refiere al código en sí hacia donde se orienta esta herramienta. Por ultimo pero no menos importante tenemos *Stack*, es una pila de servicios para la gestión y el desarrollo de proyectos software.

[1] `http://code.sidelab.es/projects/sidelab/wiki/Sidelab`

Como resultado tenemos **SidelabCode Stack** una Forja de desarrollo de aplicaciones orientada a un proceso de desarrollo.

1.2. Trabajo en TSCompany

Explicación del trabajo efectuado en TSCompany durante el desarrollo de la implementación de la Forja.

El prácticum del Máster me dio la oportunidad de entrar a trabajar en la empresa TSCompany para cubrir la plaza de becario. Desde el principio he estado interesado en la producción de software de calidad, en las herramientas e control de versiones, el desarrollo orientado a tests, la documentación y de como unificar las herramientas para obtener un rendimiento óptimo a la hora de desarrollar un proyecto, es decir, un desarrollador que desarrollar al 100 %.

Profesionalmente siempre he tenido buenos ejemplos cercanos acerca del desarrollo de software de calidad y me ha generado mucho he ido avanzando hacia

Capítulo 2

Motivación

En el mismo punto, damos la bienvenida a SidelabCode Stack, la herramienta ALM para el proceso de desarrollo.

Empezaremos con la ubicación de la forja de desarrollo SidelabCode Stack. ¿ Cual es la motivación que nos lleva a implementar un nuevo ALM ? después de haber pasado por las distintas fases como en el caso de los Frameworks.

Debido a la necesidad de tener un esqueleto para el desarrollo de un proyecto partiendo de herramientas de Software Libre actuales generando una nueva herramienta de Software Libre.

Partiendo del principio DRY (*Don't Repeat Yourself* y de la mano de *No reinventar la rueda* el proyecto surgió con la necesidad de crear una forja de desarrollo Libre y usable en distintos entornos.

Después de analizar Las posibles soluciones existentes, se optó por la unificación de las herramientas evaluadas para la generación de una nueva forja.

Después de analizar estas soluciones ALM existentes se optó por crear una nueva Forja a partir de las herramientas necesarias para el proceso de desarrollo de Software de Calidad, siguiendo las metodologías ágiles existentes.

La integración del proceso de desarrollo en la implementación de una Forja ALM es el punto clave de SidelabCode Stack.

En primer lugar se ha elaborado un esquema conceptual de la arquitectura técnica que permita la integración de los servicios, teniendo en consideración que la arquitectura de la aplicación debe lograr los siguientes objetivos:

Proveer un nivel de rendimiento suficiente para cada uno de los requerimientos de negocio o subsistemas que lo componen.

Ser escalable para cumplir las demandas de los usuarios durante el ciclo de vida de la aplicación, pudiendo absorber el impacto en el rendimiento fruto de la popularidad de la misma en el medio-largo plazo.

Ser efectiva: la modificación de una capacidad específica no requerirá el cambio de la plataforma completa, ni un cambio en el despliegue de la solución.

Ser mantenible y que sus funcionalidades sean fácilmente extensibles y reutilizables.

Permitir la fácil portabilidad de los datos de los proyectos entre forjas.

Capítulo 3

Objetivos

Constancia, metodología, facilidad de uso, herramientas al alcance de cualquier desarrollador dentro de un proceso de desarrollo para crear software de calidad.

Integrar el desarrollo completo de un proyecto:

- Creación del proyecto, usuarios, permisos y roles.
- Carpetas públicas/privadas
- Análisis de Requerimientos.
- ITS Issue Tracking System.
- Repositorio de código.
- Gestión de librerías.
- TDD Test Driven Development.
- Gestión del repositorio, parches, actualizaciones, ramas.
- Desarrollo e implementación.
- Análisis de Código.
- Despliegues en diferentes entornos.
- Integración Continua.

Generar datos aunque no se utilicen, es decir, grano gordo/grano fino.

Capítulo 4

ALM Tools

ALM Tools significa: *Application Lifecycle Management Tools*. La gestión del ciclo de vida de una aplicación, el conjunto de herramientas encargadas de guíar al desarrollador a través de un camino basado en metodologías para la creación de un Software hacia su estado del arte.

Integrar el proceso de desarrollo de Software a través de las ALM Tools como vehículo, es decir no existe en si mismo una herramienta ALM, sino que la herramienta ALM gobierna a las herramientas incluidas en el proceso de desarrollo.

4.1. Historia

En todas las empresas o comunidades que desarrollan Software siempre se aplica un proceso de desarrollo a la creación del producto. Cada una utiliza distintas herramientas para la gestión de los proyectos de Software, gestor de correo, gestor de incidencias, repositorio de código, integración continua. Pero en la mayoría de los casos de una forma dispar y sin seguir ninguna convención.

A veces el no conocimiento de otras herramientas o la no inclusión de nuevas puede hacer que el desarrollo del proyecto no mejore, partiendo de la base de que el desarrollo puede ser óptimo para las herramientas utilizadas, carece de perspectivas de mejora a corto plazo.

Si se opta por la integración de una nueva herramienta en el proceso de desarrollo el coste

de integración se habría de evaluar ya que se debería dedicar un esfuerzo a la integración ad-hoc de la nueva herramienta para el uso en este mismo entorno con el coste que conlleva, evaluación, test, integración, interoperabilidad, es decir un nuevo proyecto dentro del mismo proyecto.

Después de esta integración en el proceso de desarrollo en la empresa habría que hacer un esfuerzo para salvaguardar la información a través de las distintas herramientas por separado.

El proceso de unificación y reutilización de herramientas a los desarrolladores nos puede parecer familiar si lo comparamos con el uso de los Frameworks a principios de los años 2000. Muchas empresas o comunidades empezaron a adecuar e implementar sus desarrollos en base a un Framework creado por ellos mismos. Estos Frameworks se adecuan a sus requerimientos pero su uso era interno en la empresa y por lo tanto lejano a los estándares. Uno de los más famosos es sin duda el caso de Spring, proyecto de más de 10 años de edad que goza de buena salud y aceptación, incluso equivoca a algunos entre Java y Spring. En este pequeño paralelismo podemos encontrar el estado de las forjas de desarrollo ALM Tools, cuando el proyecto requiere de herramientas para facilitar su ciclo de vida y se van ensamblando una tras otra, que perfectamente las podemos llamar librerías, en una integración **ad-hoc** y siguiendo unos pasos repetitivos en cada nuevo proyecto, en los que humanamente todos nos podemos equivocar debido a que depende de cada uno seguir cada uno de los pasos. Estas herramientas tienden a convertirse en pequeños estándares dentro de cada grupo de desarrolladores y a repetirse en futuros proyectos, pero debido a los desarrollos **ad-hoc** carecen de escalabilidad e integración con nuevas soluciones de una forma ágil, es un escollo actualizar y por otro lado replicar un estándar para la implantación de la forja, no se tiende a dejar puertas abiertas para que más adelante el herramienta mejore. Se podría definir como uno de los casos de inanición en el desarrollo de Software o muerte por éxito.

En este punto es donde entra la famosa interoperabilidad entre las herramientas, la necesidad de interoperabilidad entre la herramientas a través de una comunicación estándar. Aquí encontramos el punto clave de las ALM Tools, la **integración de herramientas** dentro de un proceso de desarrollo.

En post de evitar la falta de replicación, además de la importancia de la interoperabilidad se ha

de tener en cuenta la replicación del contenido o la gestión de la instalación de un ALM. Siempre se ha de pensar mirando hacia adelante, no es necesario implementar las mejoras pero sí, dejar un hueco para que casen bien. Un ejemplo que puede ilustrar esta frase es la programación basada en Interfaces mediante Java, ya que las Intefaces ofrecen soluciones para implementar a medida y si se actualiza la Interfaz (en este caso es el esqueleto de la clase) para añadir una nueva funcionalidad con un método, los métodos anteriores mantienen su comportamiento dentro de cada clase que la implementa y adquieren la posibilidad de aumentar la funcionalidad implementando la nueva solución, adecuada a su entorno pero nueva, de esta forma la interoperabilidad entre las clases que utilicen esta Interfaz también se mantiene.

4.1.1. Componentes

Este conjunto de herramientas se puede dividir en varios grupos:

- Gestión de Requisitos.
- Arquitectura.
- Desarrollo.
- Test.
- Issue tracking system.
- Continuous Integration.
- Release Management.

Hoy en día las forjas ALM abundan, además de gozar de una gran popularidad entre los proyectos de Software, como podemos ver en los casos de SourceForge, Googlecode y Github (más adelante discutiremos cada proyecto). En este caso ALM Tools as a Service, debido al servicio que ofrecen, pero sólo las que son FLOSS permiten replicar ese mismo entorno en tu propia máquina, un dato muy importante a tener en cuenta, porque siempre se ha de mirar hacia adelante.

4.2. Estudio del arte de forjas

- ClinkerHQ `http://clinkerhq.com/` - Privado.
- Github - `http://github.com/` - Privado.
- SourceForge con Allura - `http://sourceforge.net/projects/allura/` - Software Libre pero incompleta (integrated Wiki, Tracker, SCM (svn, git and hg), Discussion, and Blog tools).
- Cloudbees DEV@Cloud `http://www.cloudbees.com/dev.cb` - Privado.
- CollabNet con CloudForge `http://www.cloudforge.com/` - Privado.
- Plan.io - `http://plan.io/en/` - Privado.
- Bitnami - `http://bitnami.com/` - Privado
- GForge
- Collab.net
- Google Code
- Bitbucket - https://bitbucket.org/mswlmanage2013/mswl-bitbucket-alm-tools/wiki/Home

4.3. Problemas en algunas forjas

4.4. Tablas comparativas

4.5. Conclusiones del estudio de forjas

Capítulo 5

Metodologías de desarrollo de software

5.1. Distintas metodologías

5.2. Metodología elegida

5.3. Conclusiones

Capítulo 6

Proceso de desarrollo

Descripción del proceso de desarrollo de software para crear el diseño de implementación de la forja SdelabCode Stack implantando la metodología elegida y el proceso de desarrollo por canales.

6.1. Herramientas

Capítulo 7

Diseño e Implementación

Análisis técnico de las Herramientas utilizadas.

7.1. Interoperabilidad

Interoperabilidad entre las herramientas que componen la forja. API Rest.

Capítulo 8

Pruebas y Validación

Pruebas a través de virtualización de sistemas operativos mediante herramientas libres; Vagrant, kvm, puppet.

Capítulo 9

Comunidades FLOSS

El punto diferenciador en el proyecto haciendo hincapié en la interacción con las comunidades de software de cada una de las herramientas.

Caso Real

La herramienta SidelabCode Stack se encuentra en uso para los estudiante de la UPV y dentro de la empresa TSCompany SL.

Implantación

Migración.

Metodología en uso.

Epílogo

9.1. Conclusiones

El uso de estas herramientas y su incremento de la calidad en el desarrollo, por encima de todo siendo FLOSS debido a eso la versatilidad que otorga en el momento de unificarlas en una herramienta nueva; SidelabCode Stack.

9.2. Lecciones aprendidas

Colaboración entre distintos proyectos y comunidades, interoperabilidad entre herramientas, Forjas de desarrollo y los elementos más comunes de las mismas.

9.3. Trabajo Futuro

Impulso de la comunidad a través de los canales habituales.

Integración y gestión de nuevas herramientas comunes para los desarrolladores.

Centralización de la instalación.

Apéndice A

Apéndice 1

Bibliografía

[1] Timothy Budd. *Introducción a la programación orientada a objetos*. Addison-Wesley Iberoamericana, 1994.

[2] Mark Lutz. *Programming Python*. O'Reilly, 2001.

[3] Fredrik Lundh. *Python standard library*. O'Reilly, 2001.

[4] George Reese. *Managing and using MySQL*. O'Reilly, 2002.