Automatización De La Generación De Indicadores Para El Seguimiento De Proyectos

Alicia Salamon, Patricio Maller, Alejandra Boggio, Natalia Mira, Sofía Pérez, Francisco Coenda

Departamento de Ingeniería en Informática - Instituto Universitario Aeronáutico – Av. Fuerza Aérea 6500 Córdoba – Argentina.

as.salamon, pmaller, alejandra.boggio, ncmira, sofiabeatrizperez, franciscocoenda, @gmail.com

Resumen. El presente trabajo se desenvuelve en el marco del proyecto PIDDEF 42/11 en el Departamento de Informática de la Facultad de Ingeniería del Instituto Universitario Aeronáutico. El mismo es una investigación y desarrollo acerca de los indicadores que se pueden obtener de manera automática, sin la intervención del usuario, a través de la integración de distintas fuentes de datos. Además, estos indicadores permiten, tanto al líder del proyecto y a su equipo visualizar el estado de avance de los diferentes proyectos en tiempo real poniendo a prueba una de las buenas prácticas de desarrollo de software como el "Working Agreement", como así también evaluar el desempeño del equipo y la calidad de los productos que se construyen. Esta primera experiencia se llevó a cabo en la misma institución en la que se desarrolla este proyecto.

Palabras Clave: Integración continua, Indicadores, Tablero de Mando, Dashboard, Automatización.

1 Introducción

La experiencia indica que el desarrollo de software científico es muy distinto al desarrollo de software comercial, ya que ambos poseen características particulares, por ejemplo, el dominio en el cual se desenvuelve cada uno [1]. El software generado en el ámbito científico es muy específico ya que sólo los ingenieros especialistas poseen un conocimiento acabado acerca de la problemática a solucionar.
Por mucho tiempo se ha intentado transferir las metodologías y prácticas de la ingeniería de software al ámbito científico sin los resultados esperados, ya que no se han considerado las particularidades de la comunidad científica [2]. Como consecuencia se han realizado varios estudios a nivel internacional acerca de estas cuestiones, dando como principal problemática la gestión de grandes equipos de

trabajo y otras como escasa documentación, falta de trazabilidad en los artefactos de software, etc. [3], [4].
A partir de estos antecedentes, se generó el Proyecto Piddef 42/11 titulado "Metodología y Framework de Gestión de Líneas Base de Integración de Aplicabilidad en el Desarrollo de Software para el Proyecto UAV" de la Facultad de Ingeniería del Instituto Universitario Aeronáutico.
Dentro del marco de este Piddef se realizó un Workshop con los equipos de desarrolladores de software científico-técnico de la institución, en virtud de su experiencia acumulada en este dominio. Dichos grupos conocen los procesos de desarrollo en los cuales trabajan, y por lo tanto, las situaciones críticas. El objetivo del Workshop fue elicitar las características críticas en el desarrollo de software científico-técnico, resultando en primer lugar el plano metodológico, luego gestión de proyectos, soporte y finalmente capacitación [5].
Además, se relevó que los equipos de desarrolladores adoptan pautas de trabajo en sus proyectos a partir de la Norma European Cooperation For Space Standarization (ECCS). En la sección ECCS-E-ST-40C de dicha norma se trata la disciplina "Software" que se centra en los procesos de requisitos y en los resultados esperados, haciendo hincapié en la relación sistema-software y en la verificación y validación de elementos software. Esta sección se complementa con otras: ECSS-M-ST-40 (Configuración y gestión de la información), ECSS-Q-ST-20 (Gestión de los procesos) y ECSS-Q-ST-80 (Aseguramiento de la calidad).
En respuesta a los resultados del Workshop, y a partir de los antecedentes investigados, se construyó como solución una arquitectura que implementa la integración continua [6].
El trabajo que se expone en este paper, en el marco de la arquitectura mencionada, intenta mejorar el trabajo de los equipos que desarrollan software científico-técnico, permitiéndoles visualizar un conjunto de indicadores en un tablero de mando. El mismo se puebla automáticamente a medida que el repositorio se va actualizando y el servidor de integración continua genera las construcciones. De esta manera se logra tener información en el momento oportuno a partir de los datos brindados por el servidor de integración continua Cruise Control y el repositorio Git de la arquitectura propuesta.
Las herramientas empleadas para el desarrollo de esta solución son: motor de base de datos MySQL, build manager Ant y Suite de Pentaho como herramienta de Inteligencia de Negocio (Dashboard).

2 Desarrollo

En este apartado se expone el análisis de datos fuente de Cruise Control y Git, seguidamente el diseño de la base de datos que los contendrán centralizados y relacionados entre sí, como también el diseño de los indicadores que se mostrarán en el dashboard. Finalmente se explica la construcción del tablero de mando. En la figura 1 se visualiza la arquitectura de referencia.

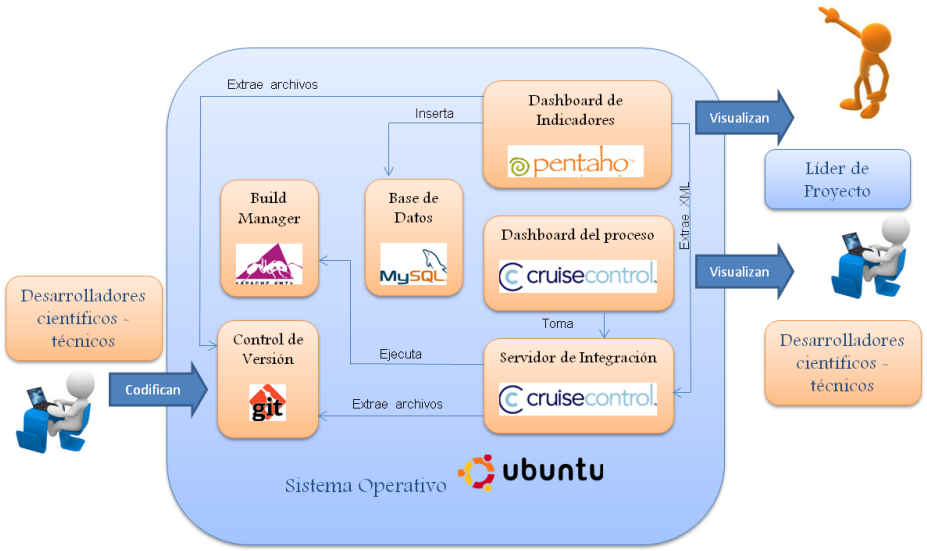

Fig. 1. Arquitectura de Referencia

2.1 Análisis de datos

En primera instancia se analizan las fuentes de datos que permiten poblar el tablero de mando; para tal actividad se extrajo una muestra de archivos XMLs arrojados por el servidor de integración Cruise Control y el log del repositorio de Git.
Se observa que los nombres de los archivos XMLs poseen una estructura específica según sea una construcción exitosa o fracasada. Por ejemplo:

Tabla 1. Nomenclatura de nombres de archivos .XML.

Caso	Nomenclatura	Ejemplo
Exito	**log***fechahora***Lbuild.***número***.xml**	log20140530111319Lbuild.1.xml
Fracaso	**log***fechahora***.xml**	log20140530134855.xml

Dentro del archivo XML cada renglón comienza con un “tag”, etiqueta o marca que delimita una región del archivo. Algunos de estos “tags” del archivo XML de Cruise Control son comunes y otros son distintos. Estos últimos son los que muestran los resultados de los “test suite” y sus respectivos “test case” en el caso del éxito, en cambio, en el fracaso se observan los “tags” “warn” y “stacktrace”.
En el análisis del log de Git se toman los siguientes atributos: hash del commit, autor, fecha y hora de la modificación, mensaje, mail del autor y fecha de creación entre otros. El log está ordenado desde las actividades más recientes hasta las más antiguas.

2.2 Diseño de la solución

En esta etapa se realiza el diseño de la base de datos con el fin de centralizar los mismos, de esta manera se logra tener datos relacionados entre sí y facilitar la obtención de indicadores.
En los siguientes párrafos se habla de "build" o "construcción", refiriéndose al producto que se obtiene tras ejecutar el "build manager" que realiza la compilación y pruebas de los componentes desarrollados.
De los XMLs provistos por Cruise Control se toman los siguientes datos relevantes acerca de cada build: nombre de la última build, nombre de la última build exitosa, fecha de la build, fecha y hora de la build, número de build (etiqueta), nombre del archivo, tiempo de limpieza, tiempo de compilación, estado de la build y proyecto al que pertenece. En caso de que la build sea exitosa se agrega los siguiente datos: tiempo de duración de la construcción de la build , tiempo de realización de los test, nombre de los test suite, nombre de los test cases, y cantidad de fallos, en cambio sí es una build fracasada se toma la cantidad de errores, el seguimiento de los pasos (tag: stacktrace) y las advertencias (tag: warn). Luego, del log de Git se extraen los siguientes datos: hash del commit, autor, fecha y hora de la modificación, mensaje, mail del autor, fecha de creación. Con la información recolectada de Cruise Control y de Git se diseña el Diagrama de Entidad-Relación de la Base de Datos.
Luego, se procede a la elaboración de los indicadores que permiten al líder y a su equipo de desarrolladores visualizar su actividad diaria haciendo énfasis en las buenas prácticas del desarrollo como por ejemplo "Working Agreement".
De acuerdo con los datos recabados en la base de datos de MySQL y retomando los resultados que se obtuvieron en el Workshop donde se determinaron necesidades como "falta de planificación de las actividades", "ausencia de procedimientos y pautas de trabajo" e "inestabilidad de plazos" se visualizan en la siguiente tabla los indicadores con su respectiva métrica y dimensión:

Tabla 2. Indicadores

Indicadores	Medida	Dimensión
Cantidad de builds por proyecto	Cantidad de builds	Por proyecto
Cantidad de builds de éxitos y fracasos por proyecto	Cantidad de builds de éxito y cantidad de builds de fracaso.	Por proyecto
Cantidad de errors y failures en el test suite por proyecto	Cantidad de errors y cantidad de failures	Por test suite y por proyecto
Cantidad de tests cases por proyecto	Cantidad de test cases	Por proyecto
Cantidad de commits por desarrollador por proyecto	Cantidad de commits	Por desarrollador y por proyecto

2.3 Desarrollo e Implementación del Dashboard de Integración

Para el desarrollo del tablero de mando se selecciona el motor de bases de datos de MySQL y para la construcción del tablero de mando la Suite Pentaho. Principalmente

fueron escogidos por ser de código abierto, por tener una gran comunidad que los mantiene y estar en continua mejora. La Suite de Pentaho posee un módulo para la realización de Dashboards denominado Ctools. Esta última facilita la creación de tableros de mando totalmente personalizados.
La Suite de Pentaho no es una sola aplicación, sino es una recopilación de programas específicos, cada uno de ellos cumplen con distintas funciones como las que se detallan a continuación:

- Informes: permite crear a los usuarios y personalizar informes además de exportar en multitud de formatos.
- Análisis: a través de las potentes herramientas ayuda al usuario a ampliar la perspectiva y mejorar la toma de decisiones en el negocio.
- Cuadros de mando: ofrece al usuario final opciones de visualización en tiempo real de los datos, gestiónanos por "displays" interactivos y últimas tecnologías multimedia para facilitar la interacción con la herramienta y manejo de datos.
- Integración de datos: dispone de aplicaciones potentes para la extracción, transformación y carga de datos en diversas plataformas de bases de datos.

Se utilizó la herramienta ETL (Extracción, Transformación, Cargar) de la Suite de Pentaho, llamada Data Integration para la elaboración de dos trabajos y once transformaciones para extraer los datos de los XMLs producidos por Cruise Control y el log de Git, luego se da el formato adecuado y por último se cargan en una base de datos relacional de MySQL de manera planificada, en función de los recursos existentes y las necesidades del líder, logrando así la automatización del proceso de integración entre el servidor de integración, el repositorio y el tablero de mando. [7].
Una vez poblada la base de datos se procede a la elaboración del tablero de mando con la herramienta Bi-server y Ctool.

3 EXPERIENCIA

La experiencia se lleva a cabo con grupos de desarrollo de software en el Instituto Universitario Aeronáutico realizando una prueba piloto con las siguientes características:
Un servidor con la siguiente arquitectura: Sistema Operativo (Ubuntu Server), Repositorio – Servidor de Control de Versión (Git), Servidor de Integración (Cruise Control), Build Manager (Ant), Motor de Base de Datos (MySQL) y Dashboard (Suite de Pentaho).
El grupo de desarrollo está compuesto por seis personas, de las cuales cinco son desarrolladores y el líder de proyecto.

3.1 Procedimiento de Integración Continua

Cada uno de los desarrolladores de los diferentes proyectos confirma su trabajo, y realiza un commit en el repositorio.

Cada cierto tiempo, que ha sido configurado en Cruise Control, el servidor de integración continua busca modificaciones en el repositorio y procede a realizar la build con los componentes necesarios de cada uno de los proyectos. Cruise Control publica la información de este proceso en el Dashboard y en la Suite de Pentaho.
Si la construcción es exitosa, se genera un producto que es almacenado en el directorio del proyecto en cuestión. Si la construcción ha sido fallida, no se obtiene ningún producto final.

3.2 Automatización de los indicadores

Las herramientas de Bi-server y Kitchen pertenecientes a la Suite de Pentaho se mantienen continuamente activas. El Kitchen de Pentaho ha sido planificado una vez al día, es decir que al final de cada día se actualiza la base de datos de MySQL que luego es consultada para la visualización de los indicadores en la el tablero de mando de la Suite de Pentaho.

3.3 Visualización del Dashboard con la Suite de Pentaho

El líder del proyecto abre el navegador web e ingresa al Bi-Server, que permite visualizar el tablero de mando con sus indicadores actualizados de manera automática y continua sin la intervención de ningún miembro del equipo. En la figura 2 se visualizan algunas de las gráficas que se muestran en el dashboard. Para simplificar el diseño no se incluyen todos los indicadores.

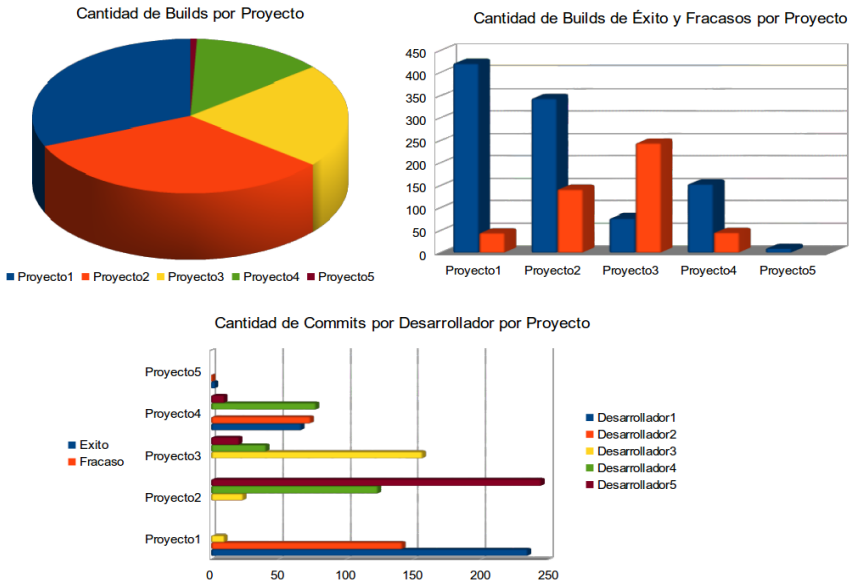

Fig. 2. Indicadores del Dashboard de integration

4 RESULTADOS

De acuerdo a la experiencia realizada, el dashboard de Pentaho muestra información acerca del estado de avance de los distintos proyectos en tiempo real, por ejemplo la cantidad de commit por proyecto realizados por cada uno de los desarrolladores y la cantidad de builds exitosas y fracasadas por proyecto entre otros. De esta manera, en caso de desvío de los objetivos propuestos por el grupo de trabajo se podrían tomar medidas correctivas optimizando así los recursos.
En cambio el dashboard de Cruise Control como se observa en la figura 3 no provee información acerca de los proyectos, sólo hace referencia a datos propios de cada una de las builds por separado, además no muestra información relacionada con el log Git.

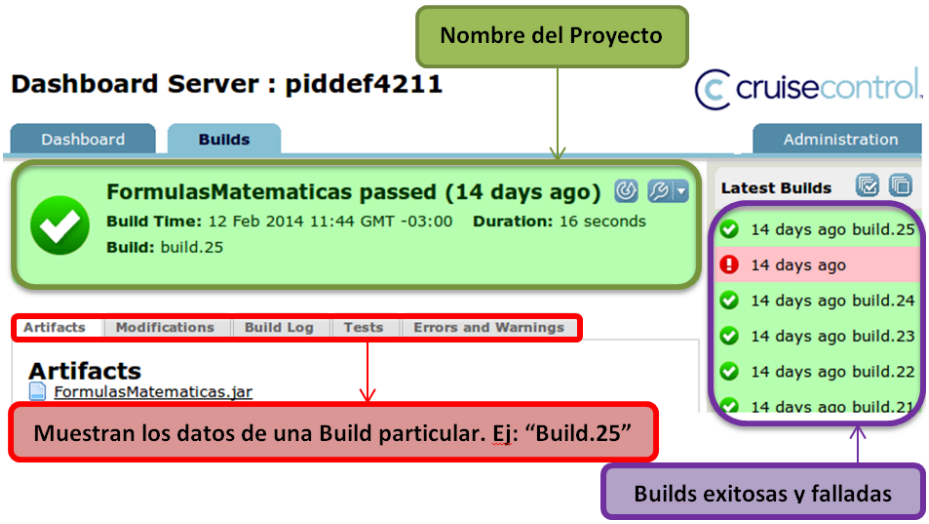

Fig. 3. Dashboard de Cruise Control

5 CONCLUSIONES

Mediante el análisis de los datos fuentes (los archivos XMLs generados por Cruise Control y el archivo log de Git), la arquitectura propuesta logra obtener indicadores que ayudan tanto al líder como al equipo de desarrolladores a realizar un seguimiento y control en tiempo real de los avances de los distintos proyectos y del desempeño de cada uno de los miembros del equipo. Además, lograr la visualización del proyecto permite que en caso de desvío se puedan tomar medidas correctivas a tiempo, aplicando de esta forma una de las buenas prácticas de desarrollo como el "Working Agreement" que promueve la eficiencia en el desarrollo de software evaluando el cumplimiento de las reglas acordadas por el grupo basándose en la visualización de los indicadores.
Lo más importante a destacar en este trabajo es la integración de las herramientas dentro de la arquitectura para la obtención de indicadores fiables de manera

automática, es decir sin la participación del usuario, haciendo que los mismos estén libres de errores, y en tiempo real, logrando así el aprovechamiento al máximo de las fuentes de datos existentes.
La investigación actual en el dominio continúa, apunta a seguir evaluando la posibilidad obtener indicadores desde otras fuentes de la arquitectura propuesta.

REFERENCIAS

1. Segal J., "Scientists and software engineers: a tale of two cultures." in PPIG University of Lancaster, UK, 2008.
2. Basili V. et al., "Understading the High-Performance-Computing Community: A Software Engineer's Perspective" in IEEE Software Computer Society, vol. vol. 25, no. 4, Los Alamitos, CA, USA, 2008, pp. pp. 29-36.
3. Kelly D., "A Software Chasm: Software Engineering and Scientific Computing." in IEEE Computer Society, Los Alamitos, CA, USA, 2007.
4. Hannay J. E., MacLeod C., and Singer J., "How Do Scientists Develop and Use Scientific Software?" in IEEE Computer Society, Washington, DC, USA, 2009.
5. Maller P., Salamon A., Mira N., Boggio A., Giró J., Cuozzo J., Perez S., Coenda F. "New Practices to Structure and Elicit Improvement Opportunities in Scientific Software Development Teams. GDN 2012. Submitted for publication.
6. Duvall P. M., Matyas S., Glover A. "Continuous Integration: Improving Software Quality and Reducing Risk"Addison-Wesley, Boston, 2007.
7. Roland B., Jos Van D. "Pentaho Solutions: Business Intelligence and Data Warehousing with Pentaho and MySQL". Wiley. Indianápolis, Indiana, USA, 2009.