Desarrollo ágil y reutilización de Software Libre

Nancy Fernández[1], Carina Reyes[2], Jorge Ramirez[3], Nilsa Sarmiento Barbieri[4]
[1,2,3,4] Facultad de Ciencias Exactas. Universidad Nacional de Salta
[1]{nancy.fernandezg13@gmail.com, [2] reyescarina@gmail.com,[3]jorge@dragonlibre.net,
[4]nilsamsarmiento@gmail.com

Resumen. Las metodologías ágiles de desarrollo de software enfatizan la importancia del "software funcionando" por sobre la documentación extensiva. Al mismo tiempo, la posibilidad de reutilización que ofrecen las licencias de software libre y la amplia oferta existente de productos bajo ese tipo de licencias invitan a explorar las ventajas y dificultades que puede acarrear su adopción en el marco de diversos procesos de metodologías ágiles. Mediante una experiencia de desarrollo parcial, detectamos posibilidades y dificultades de un abordaje conjunto.

Palabras Clave: Software Libre, reutililzación, metodologías ágiles.

1 Introducción

El desarrollo de software orientado a su utilización en ámbitos de definición de políticas públicas o para la toma de decisiones en contextos comunitarios, requiere de que el mismo no resulte subordinado a las exigencias de una entidad privada; al mismo tiempo, es conveniente que el mismo pueda ser analizado y evaluado de manera independiente por diferentes actores, los que no necesariamente forman parte del proceso de desarrollo.

Estos atributos son compatibles con el Software Libre, el que a su vez ofrece una alternativa para la reutilización definida desde las licencias de distribución que las caracterizan; al mismo tiempo, estos productos ofrecen una mayor transparencia, al estar su código fuente a disposición de quienes deseen analizarlo o modificarlo. Estas ventajas, sin embargo, se ven limitadas por la insuficiencia de documentación de muchos proyectos de Software Libre, así como la carencia de lineamientos que orienten la reutilización sistemática en el marco de metodologías de desarrollo arraigadas en la ingeniería de software.

Por otra parte, las metodologías ágiles surgieron a partir de la necesidad de atender desarrollos con requerimientos cambiantes y demandas disímiles de parte de los destinatarios finales, por lo que su adopción puede resultar beneficiosa para desarrollos en ámbitos dependientes del Estado, como es el caso de las Universidades

Nacionales. Al mismo tiempo, este tipo de metodologías privilegia “software funcionando por sobre documentación excesiva” [1].

A fin de explorar las ventajas y las dificultades que pueden presentarse en la incorporación de la reutilización de software libre en el marco de una metodología ágil de desarrollo, realizamos una experiencia a partir de los requerimientos específicos enmarcados en un proyecto de investigación.

Nos propusimos detectar ventajas y desventajas de incorporar la reutilización de software libre durante el proceso de desarrollo, elaborando hipótesis que orienten futuras investigaciones.

Para ello, llevamos adelante un desarrollo experimental tomando como referencia a un proceso característico de una metodología ágil: realizamos el desarrollo parcial de una aplicación guiándonos por el enfoque Feature Driven Development; la tarea realizada fue similar a la correspondiente a una iteración del proceso de desarrollo, en la que se definieron al comienzo de la misma una serie de funcionalidades que deberá cumplir el producto al final de la fase. No se trató de una simulación automatizada, sino de un desarrollo efectivo orientado a detectar las dificultades y las ventajas que se adviertan en el proceso, en la búsqueda de resultados cualitativos y orientativos de futuras experiencias.

El conjunto de funcionalidades a desarrollar fue seleccionadas de acuerdo con las necesidades concretas del proyecto de investigación 2427 del CIUNSa, referidos al desarrollo de GIS para el soporte en la toma de decisiones en la planificación en energías renovables mediante la reutilización de software libre. La elección de un dominio como el GIS ofrece la posibilidad de acotar las funcionalidades requeridas, al tiempo que se cuenta con algunas propuestas de clasificación [2] de piezas de software específicas para este dominio.

El presente trabajo se organiza de la siguiente forma: en la sección siguiente repasamos trabajos relacionados con nuestra exploración; a continuación, presentamos las características del experimento realizado; la siguiente sección discute el desarrollo de la experiencia; a continuación, se señalan las limitaciones de la experiencia realizada; y finalmente se presentan las conclusiones y los trabajos futuros.

2 Trabajos relacionados

Muchos autores han explorado las similitudes entre las metodologías ágiles y procesos o métodos habituales en el desarrollo de Software Libre. Russo y otros publicaron un libro dedicado al tema [3]. Abrahamsson y otros[4] incluyeron al Software Libre y Código Abierto en su revisión de metodologías ágiles.

En cuanto a la relación entre metodologías ágiles y reutilización, Turk y otros [5] señalaron que entre las limitaciones de los procesos de estas metodologías; los autores advierten que éstas apuntan más a resolver una situación específica, antes que a proponer soluciones de carácter más general que pudieran servir de base para la reutilización.

Kircher y Hofman [6] describieron los resultados de una experiencia que combinó reutilización y metodologías ágiles, la cual se desarrolló en el marco de la firma Siemens Healthcare; ésta adoptó desde hace varios años la Ingeniería de Software de Línea de Productos para aprovechar las funcionalidades comunes en diferentes líneas de producto. Los autores describen de qué forma se emplearon métodos ágiles para el desarrollo de artefactos a ser incluidos en la Plataforma de la Línea de Productos[7] correspondiente, así como los desafíos y dificultades de ese proceso.

La utilización de metodologías ágiles en el ámbito específico de desarrollos para el Estado fue explorado en diversos trabajos: a modo de ejemplo, Upender refiere la incorporación de Scrum para mejorar la entrega de software en proyectos gubernamentales [8] , y Van Velsen y otros [9] abordan el desarrollo de sistemas para el Instituto Nacional de Salud de Estados Unidos (NIH por sus siglas en inglés) en un contexto de multidisciplinariedad y requerimientos heterogéneos, en los cuales resultan de interés la aplicación de prácticas y métodos provenientes del mundo “ágil”.

3 Objetivos

Las siguientes preguntas orientaron el desarrollo general del presente trabajo:

¿Es factible incorporar de manera sistemática la reutilización de software libre en el marco de una metodología ágil?

Tal incorporación ¿qué ventajas presentaría respecto de la reutilización oportunista?

¿De qué manera, qué roles tendrían la responsabilidad de definir qué se reutiliza?

¿Con qué herramientas deberían contar las y los desarrolladores para facilitar la reutilización de software libre?

Teniendo en cuenta esos interrogantes, este trabajo busca responder:

¿Qué dificultades surgieron para la reutilización? ¿De qué manera afecta al desarrollo de la iteración? Esta última pregunta puede especificarse: ¿La fase definida en la metodología contempla alguna instancia para la reutilización o es necesario adaptarla? ¿las responsabilidades de los distintos roles se adecuan a las tareas que requiere la reutilización? ¿Qué restricciones surgen de la adopción de elementos reutilizables para las etapas posteriores del desarrollo?

4 Desarrollo de la experiencia

El desarrollo realizado tomó como referencia la metodología Feature Drivement Development. A continuación, presentamos una breve descripción de la misma, destacando las fases que prevé.

4.1 Feature-Driven Development

La metodología Feature-Driven Development (FDD) [10] [11] permite crear software mediante un ciclo de vida iterativo e incremental a través de prácticas enfocadas en la perspectiva de las funcionalidades que poseen valor para el cliente. Considera la figura del jefe de proyecto y una fase de arquitectura. También está dirigida por modelos, y es de iteraciones cortas.

FDD define 5 procesos: Proceso 1- Desarrollar el modelo global (Develop overall model), Proceso 2- Elaborar lista de funcionalidades (Build feature list), Proceso 3- Planificar funcionalidades(Plan by feature), Proceso 4- Diseñar funcionalidades (Design by feature) y Proceso 5- Desarrollar funcionalidades (Build by feature). Los tres primeros procesos permiten que los equipos de desarrollo tomen un tiempo previo para arrancar, montar entornos, planificar, e incluso hacer un estudio previo de la arquitectura.

4.2 Desarrollo de la experiencia

La actividad se concentró fundamentalmente en los procesos 4 y 5, que son los que se efectúan de manera iterativa, aportando el incremento de las funcionalidades del producto en desarrollo.

En los tres primeros procesos previstos en la metodología, se estableció que el proyecto tiene la necesidad de registrar información geográfica mediante la identificación de zonas factibles de instalación de paneles solares y molinos, junto a una descripción de la misma. Esta descripción puede consistir en texto y/o imágenes de la zona. Luego se elaboró un listado de las funcionalidades necesarias y se planificaron considerando la reutilización de software.

La búsqueda de un componente libre se orientó hacia un software que permitiera la visualización de mapas, que pudiera presentar a la ciudad de Salta y sus alrededores con niveles variables de detalle. El proceso de selección tomó en consideración la documentación de los componentes y la disponibilidad de ejemplos de uso.

Aquí surge una primera observación: es preciso definir cuánto tiempo se destinará a la búsqueda de oportunidades de reutilización, a partir de lo cual recién será posible continuar con el desarrollo. Sería necesario, además, definir quién o quiénes tendrían

la responsabilidad de decidir sobre la conveniencia o no de reutilizar, y qué componente en concreto se utilizaría.

Al tratarse de un desarrollo de tipo experimental, no planteamos a priori restricciones en cuanto a la licencia del software a buscar.

Utilizamos un buscador general y relevamos sitios sobre Sistemas de Información Geográfica con código abierto (https://www.gislounge.com/open-source-gis-applications/, https://spatial.usc.edu/software/open-source-gis-applications/, http://www.geolatte.org/)

En el proceso, tomamos en cuenta la disponibilidad de ejemplos de uso, documentación y la utilización en otros proyectos. No existe un repositorio o catálogo de componentes que simplifique esta tarea de búsqueda.

Las dificultades surgidas en esta etapa podrían reducirse contando con repositorios o catálogos orientados a facilitar la reutilización [12]. Cabe señalar que en el marco del proyecto de investigación en el que participan los autores está en proceso de construcción una herramienta de ese tipo.

A partir de esas consideraciones, seleccionamos un componente en Java con licencia GPL: JMapViewer.

Esa decisión trae dos consecuencias para el desarrollo ulterior no sólo de la iteración bajo análisis sino también para las etapas subsiguientes: el software desarrollado deberá distribuirse bajo licencia como un todo bajo la licencia GPL y se establecen restricciones en cuanto al lenguaje de programación.

Ya en el trabajo con el componente, encontramos dos formas en que se lo puede reutilizar, cada una de las cuales tendrá consecuencias en la arquitectura: como un componente de “caja negra”, que permite un menor acoplamiento, o modificando el propio componente para obtener los resultados esperados. Esta decisión también puede traer aparejadas condicionantes para las etapas futuras. En el caso concreto de la metodología que se toma como referencia, se advierte la necesidad de que algunas decisiones de diseño sean adoptadas con anterioridad a las etapas iterativas.

La primera opción requiere escribir un hijo de la clase `DefaultMapController`, a la que se le sobreescriban los métodos que se requieren (por ejemplo, `MouseClicked`). La segunda posibilidad consiste en modificar directamente el código original, cambiando el comportamiento del componente.

La experiencia culminó con el desarrollo de un producto capaz de mostrar las funcionalidades esperadas, al tiempo que establece restricciones para etapas posteriores.

La figura 1 muestra una captura de pantalla incorporando el componente JMapViewer configurado para mostrar la zona de interés. La captura del click simple del mouse se realizó modificando el código original, preservando las respuestas originales a otros eventos del mismo.

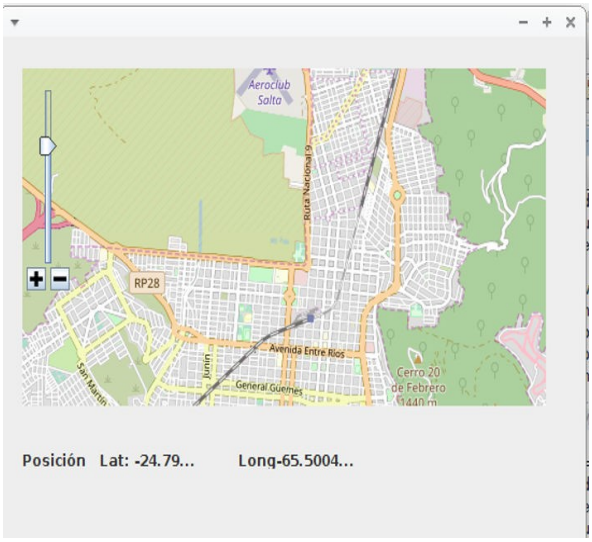

Fig. 1. Captura de la pantalla de la aplicación en desarrollo

5 Limitaciones del presente trabajo

La experiencia realizada permite detectar algunas particularidades de interés respecto de la adopción de la reutilización de software libre en un marco de desarrollo ágil; no obstante, es preciso señalar que existen particularidades que comprometen la posibilidad de generalizar los resultados.

El equipo que trabajó en el desarrollo está conformado por integrantes de un proyecto de investigación con cierta experiencia en Sistemas de Información Geográfica y desarrollo de Software Libre; los plazos de realización de la experiencia no necesariamente se condicen con circunstancias de producción, aunque tampoco sean extrañas a numerosos desarrollos en el contexto de proyectos de investigación en Universidades Públicas. A nuestro juicio, para desarrollos en contextos similares, las observaciones surgidas de esta experiencia pueden constituir una guía, aunque serán necesarias otras experiencias para extraer conclusiones más generales.

6 Conclusiones

Esta experiencia muestra la factibilidad de desarrollar de forma rápida mediante la reutilización de Software libre. No obstante, la incorporación sistemática de la misma

en desarrollos ágiles puede requerir adaptaciones de los procesos propios de esas metodologías.

De la experiencia realizada surge la necesidad de que la decisión de reutilizar Software Libre sea previa a los procesos iterativos en los que se desarrolla el software de manera incremental.

Tal definición debe contemplar explícitamente la o las licencias aceptables para la selección de componentes libres, las que están condicionadas por la licencia del Software a desarrollar, al tiempo que incide en las licencias de otros componentes eventualmente reutilizados.

En el caso de FDD, estas decisiones deberán adoptarse en las primeras etapas del proyecto, quedando incorporadas en la (o las) licencias a adoptar y en los lineamientos arquitectónicos.

La reutilización puede ser especialmente útil para lograr resultados rápidos que ofrezcan funcionalidades de interés para los destinatarios del desarrollo de software. En nuestro caso, el gráfico 2 muestra la integración de la reutilización en las dos fases que integran las iteraciones en la metodología FDD.

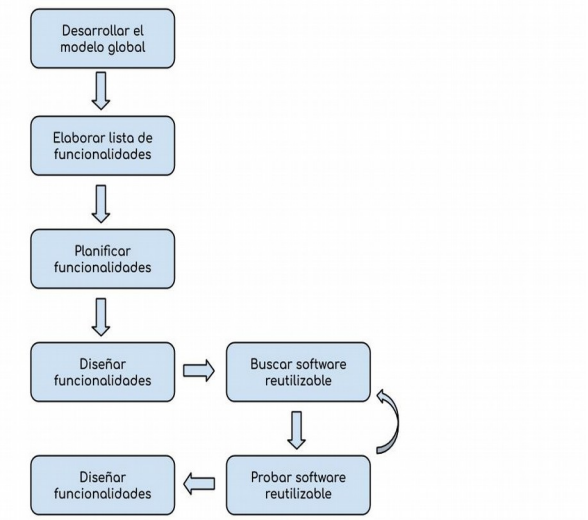

Fig. 2. Una posible adaptación de FDD para contemplar la reutilización sistemática.

En futuros trabajos nos proponemos analizar la factibilidad de incorporar la reutilización de Software libre bajo diferentes metodologías ágiles, de modo de

posibilitar comparaciones, con miras a elaborar propuestas de aplicación o adaptación de las mismas.

Para avanzar hacia la enunciación de lineamientos orientativos, será preciso cotejar diferentes experiencias en contextos diferentes. En particular, está previsto trabajar con estudiantes de la carrera de Análisis de Sistemas de la UNSa para contar con diferentes configuraciones en experiencias futuras.

Referencias

1. Beck, K., Cockburn, A., Fowler, M.: Manifiesto por el Desarrollo Ágil de Software, https://agilemanifesto.org/iso/es/manifesto.html.
2. Gaetan, G., Martin, A., Molina, S., Saldaño, V.E., Bucella, A., Cechich, A.: Publicación y Selección de Componentes para SIG. In: XII Workshop de Investigadores en Ciencias de la Computación (2010).
3. Russo, B.: Agile technologies in open source development. IGI Global (2009).
4. Abrahamsson, P.: Agile Software Development Methods: Review and Analysis (VTT publications). (2002).
5. Turk, D., France, R., Rumpe, B.: Limitations of agile software processes. arXiv preprint arXiv:1409.6600. (2014).
6. Kircher, M., Hofman, P.: Combining systematic reuse with Agile development: experience report. In: Proceedings of the 16th International Software Product Line Conference-Volume 1. pp. 215–219. ACM (2012).
7. Metzger, A., Pohl, K.: Software product line engineering and variability management: achievements and challenges. In: Proceedings of the on Future of Software Engineering. pp. 70–84. ACM (2014).
8. Upender, B.: Staying agile in government software projects. In: Agile Development Conference (ADC'05). pp. 153–159. IEEE (2005).
9. Van Velsen, L., Wentzel, J., Van Gemert-Pijnen, J.E.: Designing eHealth that matters via a multidisciplinary requirements development approach. JMIR research protocols. 2, e21 (2013).
10. Marek Rychlý, P.T.: A tool for supporting feature-driven development. Balancing Agility and Formalism in Software Engineering, Springer Berlin Heidelberg, pp. 196-207, 2008.
11. J. Pang, L.B.: Refining Feature Driven Development - A methodology for early aspects. Early Aspects 2004, AOSD'04, Lancaster, UK, March 2004.
12. Ramirez, J., Gil, G., Massé Palermo, M.L.: Hacia un catálogo práctico de componentes de F/OSS para la reutilización. In: 41 Jornadas Argentinas de Informática. pp. 45–52. , La Plata (2012).