IV Jornadas de Calidad de Software y Agilidad

JCSA 2021

12 y 13 de noviembre de 2021

IV JORNADAS DE CALIDAD DE SOFTWARE Y AGILIDAD

IV

LIBRO DE ACTAS

UNIVERSIDAD NACIONAL DE MISIONES

UTN

Libro de Actas de las Cuartas Jornadas de Calidad de Software y Agilidad / Gladys Noemí Dapozo, Emanuel Irrazábal, María de los Ángeles Ferraro, Horacio D. Kuna, Eduardo Zamudio, Alice Rambo, César Acuña, Verónica Bollati y Noelia Pinto ; compilación de Gladys Noemí Dapozo ; Emanuel Agustín Irrazábal. - 1a ed compendiada. - Corrientes : Universidad Nacional del Nordeste. Facultad de Ciencias Exactas, 2021. Libro digital, PDF

Editorial de la Universidad Nacional del Nordeste (EUDENE)

ISBN 978-987-3619-72-4

1. Software. 2. Jornadas. 3. Argentina. I. Dapozo, Gladys Noemí, comp. II. Irrazábal, Emanuel Agustín, comp.

CDD 004.0711

Fecha de catalogación: 03/01/2022

Autoridades

Universidad Nacional del Nordeste

Rectora: Prof. María Delfina Veiravé
Vicerrector: Dr. Mario H. Urbani

Facultad de Ciencias Exactas y Naturales y Agrimensura

Decana: Mgter. María Viviana Godoy Guglielmone
Vicedecano: Dr. Enrique Laffont

Universidad Nacional de Misiones

Rectora: Mgter. Alicia Violeta Bohren
Vicerrector: Ing. Fernando Luis Kramer

Facultad de Ciencias Exactas, Químicas y Naturales

Decano: Dr. Luis Brumobsky
Vicedecano: Dr. Marcelo Marinelli

Universidad Tecnológica Nacional

Rector: Ing. Héctor Aiassa
Vicerrector: Ing. Haroldo Avetta

Facultad Regional Resistencia

Decano: Jorge A. De Pedro
Vicedecano: Dr. Ing. Walter Gustavo Morales

IV Jornadas de Calidad de Software y Agilidad

Este evento es organizado en forma conjunta por las universidades de la región que tienen unidades académicas que ofrecen carreras de Informática, UNNE, UTN y UNaM, a través de un acuerdo específico institucional, con el objetivo de difundir avances significativos en el campo de conocimiento de la Ingeniería de Software, Calidad y Agilidad; y propiciar el encuentro entre las universidades, las empresas y los organismos del Estado para contribuir al desarrollo de la industria del software en la región de influencia de las universidades participantes. Las Jornadas de Calidad de Software y Agilidad (JCSA) se inician en el año 2017, bajo el nombre original de Jornadas de Calidad de Software, en esta edición, las jornadas se consolidaron como un foro regional de referencia, ampliando su espectro para enfatizar también temas relacionados al uso de la agilidad. Además, se propuso la publicación de los trabajos académicos presentados, previa evaluación de pares, que se incorporan en este libro de actas. Durante el evento se realizaron tres talleres vinculados con los temas de la Jornada, se expusieron los artículos y posters académicos aceptados y se presentaron experiencias de la industria del software, con la participación de empresas y organismos del Estado. La conferencia inaugural denominada "Estrategias pruebas de aceptación para Entrega Continua", estuvo a cargo de Diego Fontdevilla (UNTREF). Las actividades propuestas estuvieron destinadas a ingenieros y licenciados en sistemas, estudiantes y docentes de estas especialidades, profesionales y empresarios del sector del software y servicios informáticos, así como también público interesado en la temática. En total participaron más de 200 personas en las actividades programadas, lo que evidencia el interés que estos temas suscitan en los destinatarios.

Comité Organizador

Mgter. Gladys Noemí Dapozo
(FaCENA - UNNE)

Dr. Emanuel Irrazábal
(FaCENA - UNNE)

Lic. María de los Ángeles Ferraro
(FaCENA - UNNE)

Dr. Horacio D. Kuna
(FQCEyN - UNaM)

Dr. Eduardo Zamudio
(FQCEyN - UNaM)

Ing. Alice Rambo
(FQCEyN - UNaM)

Dr. César Acuña
(FRRe - UTN)

Dra. Verónica Bollati
(FRRe - UTN)

Dra. Noelia Pinto
(FRRe - UTN)

Comité de Programa

Mgter. Cristina Greiner
(GICS - FaCENA - UNNE)

Mgter. Gladys Noemí Dapozo
(GICS - FaCENA - UNNE)

Dr. Emanuel Irrazábal
(GICS - FaCENA - UNNE)

Dr. Rubén Bernal
(GICS - FaCENA - UNNE)

Dr. David la Red Martínez
(FaCENA - UNNE)

Dra. Sonia Mariño
(FaCENA - UNNE)

Mgter. M. Viviana Godoy Guglielmone
(FaCENA - UNNE)

Mgter. Mónica Tugnarelli
(FCAD - UNER)

Dra. Gabriela Arévalo
(DCyT - UNQ)

Dra. María Fernanda Golobisky
(UTN - FRStaFe)

Master Ariel Pasini
(LIDI - UNLP)

Mgter. Pablo Thomas
(LIDI - UNLP)

Dr. Fernando Emmanuel Frati
(DCByT - UNdeC)

Dra. Marcela Genero Bocco
(Grupo Alarcos - UCLM)

Dr. Jorge Andrés Diaz Pace
(ISISTAN - CONICET)

Dr. Nazareno Aguirre
(FCEFQyN - UNRC / CONICET)

Dra. Nancy Ganz
(IIDII-FCEQyN - UNaM)

Esp.Ing.Alice Rambo
(IIDII-FCEQyN - UNaM)

Ing. Selva Nieves Ivaniszyn
(FCEQyN - UNaM)

Lic. Sergio Caballero
(FCEQyN - UNaM)

Lic. Martín Rey
(IIDII-FCEQyN - UNaM)

Dr.Eduardo Zamudio
(IIDII - FCEQyN - UNaM)

Dr.Horacio Kuna
(IIDII - FCEQyN - UNaM)

Dr. Diego Godoy
(UGD)

Ing. Edgardo Belloni
(UGD)

Mag. Liliana Cuenca Pletsch
(CINApTIC - UTN - FRRe)

Dr. César J. Acuña
(CINApTIC - UTN - FRRe)

Dra. Verónica Bollati
(CINApTIC - UTN - FRRe / CONICET)

Dra. Noelia Pinto
(CINApTIC - UTN - FRRe)

Esp. Gabriela Tomaselli
(CINApTIC - UTN - FRRe)

Ing. Nicolas Tortosa
(CINApTIC - UTN - FRRe)

Ing. Valeria Sandobal Verón
(GIESIN- UTN - FRRe)

Ing. Germán Gaona
(CINApTIC - UTN - FRRe)

Dr. Horacio Leone
(INGAR - UTN - FRSF)

Dr. Silvio Gonnet
(INGAR - UTN - FRSF)

Dr. Gustavo Rossi
(LIFIA - UNLP)

Dra. Alejandra Garrido
(LIFIA - UNLP)

Dr. Andrés Rodriguez
(LIFIA - UNLP)

Dr. Marcelo Estayno
(UNSAM)

Dr. Luis Olsina
(GIDIS - UNLPam)

Dra. Luciana Ballejos
(CIDISI - UTN - FRSF)

Dra. Luciana Roldan
(INGAR - UTN - FRSF)

Dra. Milagros Gutierrez
(UTN - FRSF)

Dra. Mariel Alejandra Ale
(CIDISI - UTN - FRSF)

Dra. Elsa Estevez
(UNSur / CONICET)

Dra. Alicia Mon
(UNLaM)

Sonar JUploader, aplicación para el análisis, sincronización y actualización automática de proyectos a Sonar Cloud

Juan Alberto Pinto Oppido, Martin Cardozo, Juan Andrés Carruthers, Emanuel Agustín Irrazábal

Grupo de Investigación en Calidad de Software, Facultad de Ciencias Exactas y Naturales y Agrimensura, Universidad Nacional del Nordeste
juan_al_pinto@hotmail.com
{mcardozo, jacarruthers, eirrazabal}@exa.unne.edu.ar

Resumen. Sonar Cloud es una plataforma que permite el monitoreo del código fuente verificando el cumplimiento de reglas, detección de defectos en el código, definición del nivel de cobertura, entre otras. Cada una de estas características en el código fuente son cuantificadas o medidas por medio de un conjunto métricas, y Sonar Cloud ofrece una herramienta llamada Sonar Scanner que puede integrarse a otros desarrollos para implementar funcionalidades de análisis estático de código con la plataforma. Sin embargo, esta herramienta no está diseñada para analizar un lote de proyectos automáticamente, siendo esto un problema al momento de generar métricas para estudios empíricos del software. De esta manera surge la aplicación Sonar Juploader que busca brindar estas facilidades a los grupos de investigación. En el siguiente artículo se detalla Sonar Juploader, una aplicación java que permite analizar automáticamente un lote de proyectos con Sonar Cloud. La aplicación da soporte a la gestión y uso de organizaciones de Sonar Cloud, pre-configuración de proyectos, análisis de proyectos, y visualización de reportes.

Palabras Clave: Sonar Cloud, Análisis lotes de proyectos, Herramienta, Análisis estático código fuente.

1 Introducción

En la Ingeniería de Software se trabaja mayoritariamente con la "construcción de aplicaciones software multi-versión" [1]. Por lo tanto, muchas de las actividades asociadas con una aplicación software provocan revisiones para mejorar la funcionalidad o para corregir errores, especialmente en las metodologías ágiles [2]. En el caso del desarrollo software, la calidad puede estudiarse desde el punto de vista de: i) la calidad del proceso de desarrollo software, y ii) la calidad del código fuente [3]. En este último caso es necesario obtener métodos empíricos para demostrar la calidad del software [4] y utilizar evidencia directamente relacionada con el producto software resultante a partir de métricas e indicadores que se vinculen directamente con la calidad [5].

El uso masivo de repositorios para el código fuente (por ej., SourceForge, GitHub o Maven) le ha otorgado a los investigadores e ingenieros de software el acceso a millones de proyectos y, por lo tanto, datos para el desarrollo de estudios empíricos [6 - 8]. En este contexto, una práctica para demostrar la efectividad de las métricas como predictores de las características de calidad del software es la construcción de las denominadas colecciones de proyectos [9]. Estas colecciones son un insumo para los grupos de trabajo y sirven como mecanismo de comparación para distintos tipos de experimentos.

Sin embargo, dada la gran cantidad de proyectos que pueden estar contenidos en estas colecciones, la generación de métricas no resulta una tarea trivial. Aunque existan herramientas que facilitan el análisis automático del código fuente, este solamente es aplicado a un proyecto a la vez. Por ello es necesaria una herramienta que de soporte al proceso de análisis de una colección masiva de proyectos software.

Sonar JUploader es una aplicación java, cuya función principal es analizar lotes de proyectos de forma automática con Sonar Cloud, evitando la tarea manual al usuario de realizar numerosos análisis individuales, asistiendo en la pre-configuración de los mismos y obteniendo reportes y gráficos estadísticos en base a los análisis realizados.

El resto del artículo se encuentra organizado de la siguiente manera. En la sección 2, están los trabajos relacionados. En la sección 3 y 4 se presenta la aplicación desarrollada y su evaluación en un lote de proyectos respectivamente. Finalmente, en la sección 5 se encuentran las conclusiones del trabajo desarrollado.

2 Trabajos relacionados

En los últimos años un gran número de herramientas para el análisis estático del código fuente han sido reportadas en estudios empíricos para la recolección de metadatos de proyectos software [10 – 12]. Estos metadatos en forma de métricas o indicadores revelan diferentes características del código fuente desde su tamaño, complejidad, acoplamiento, cohesión, entre otras [13]; proporcionando información acerca de la calidad del software en cuestión.

Algunos ejemplos de herramientas bastante recurrentes en la literatura para la generación de metadatos son CKJM [14] que calcula 8 métricas orientadas a objetos con el código compilado en Java. PMD [15] que evalúa la calidad del código por medio de distintas reglas reportando fallas en el código y métricas de diseño. Findbugs (ahora SpotBugs) [16] utiliza análisis estático para buscar bugs en el código Java. Sonar Qube o Cloud en sus versiones local y remota respectivamente [17], es una solución basada en la nube para análisis continuo de la calidad, fiabilidad y seguridad del código, detectando defectos, vulnerabilidades y errores.

Sin embargo, estas herramientas en su mayoría, carecen de funcionalidades que permitan su ejecución de manera iterativa a un lote de proyectos, siendo esto un limitante al momento de generar metadatos necesarios para realizar estudios empíricos del software. De esta manera surge la aplicación Sonar Juploader que busca brindar estas facilidades a los grupos de investigación.

3 Sonar Juploader

Sonar Juploader es una aplicación de escritorio construida con el lenguaje de programación java, que emplea la plataforma de análisis de Sonar Cloud a través de una herramienta a nivel de comandos, llamada Sonar Scanner [18]. Sonar Scanner puede ser utilizada para realizar el análisis del código fuente de un proyecto tras configurar y especificar manualmente una serie de parámetros del mismo.

3.1 Sonar Scanner

Sonar Cloud brinda una serie de herramientas para realizar escaneo y análisis de proyectos en base al sistema de construcción utilizado en el mismo (Sonar Scanner for Gradle, Sonar Scanner for Maven, Sonar Scanner for Ant) o la plataforma/herramienta de automatización y gestión usada (Sonar Scanner for .Net, Sonar Scanner for Jenkins, Sonar Scanner for Azure Devops). Para realizar un análisis genérico que no se apoye en ninguna de estas tecnologías específicas, existe Sonar Scanner, el cual se adaptará al tipo de proyecto según el lenguaje de programación presente en los ficheros de código fuente.

Los requisitos mínimos para realizar un análisis con Sonar Cloud son los siguientes:

- Una cuenta en https://sonarcloud.io/ donde crear una organización para subir los análisis realizados, y generar tokens de seguridad usados para obtener acceso a la organización desde fuera por la aplicación.
- Un proyecto software a analizar, previamente configurado con su respectivo fichero "sonar-project.properties" el cual contiene la clave del proyecto dentro de la organización, el nombre del proyecto, su versión, su lenguaje, la ubicación de sus ficheros de código fuente, la ubicación de sus ficheros compilados y la ubicación de sus librerías, entre otros datos.
- Contar localmente con la herramienta Sonar Scanner obtenida desde la web oficial desde https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/

3.2 Administración de organizaciones

Con la administración de organizaciones de Sonar Cloud el usuario es capaz de cargarlas, editarlas y borrarlas, y serán usadas a la hora de realizar análisis para evitar la carga reiterada de la misma información en cada oportunidad. En la Fig. 1 puede observarse la interfaz de administración de organizaciones, donde cada una posee un título identificatorio para facilitar su selección, una descripción para comprender su finalidad, el nombre clave dentro de la plataforma Sonar Cloud, su respectivo token de seguridad para el acceso, y la carpeta desde la cual se obtienen los proyectos a analizar, en caso de que esta no varíe, de lo contrario puede omitirse este dato.

4

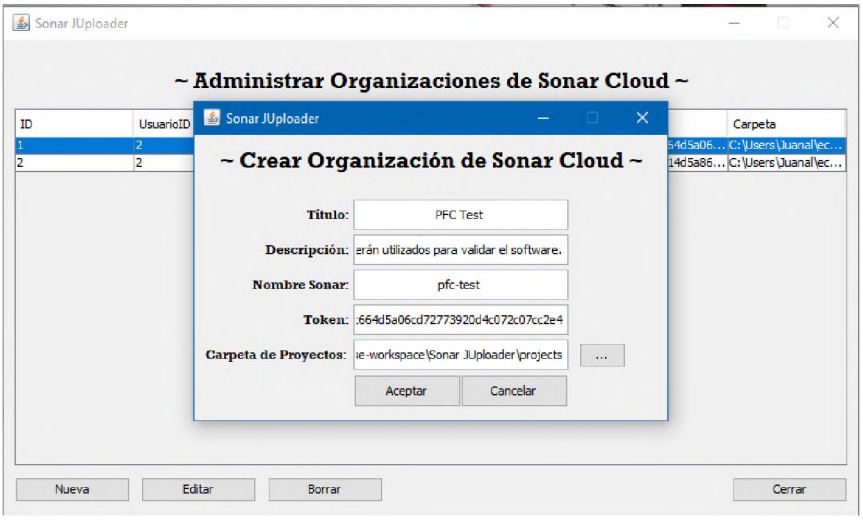

Fig. 1. Interfaz de creación de organizaciones de Sonar JUploader.

3.3 Pre-configuración de proyectos

La pre-configuración de proyectos de Sonar Cloud requiere la creación del fichero "sonar-project.properties", ubicado en la raíz del directorio del proyecto, previo a la realización de un análisis. En la Fig. 2 podemos observar la interfaz de pre-configuración donde podremos elegir una de las organizaciones creadas, y automáticamente se obtendrá la descripción y se generará el primer segmento del ProjectKey que la referencia. Esto se omite si se ingresa como invitado a la aplicación, debiendo editar la ProjectKey manualmente al no contar con organizaciones cargadas.

Luego, tras elegir la dirección donde se encuentra el proyecto a pre-configurar, el nombre de dicho proyecto se actualizará automáticamente en el ProjectKey y ProjectName, pero podemos editarlos si deseamos ajustarlos a otra estructura.

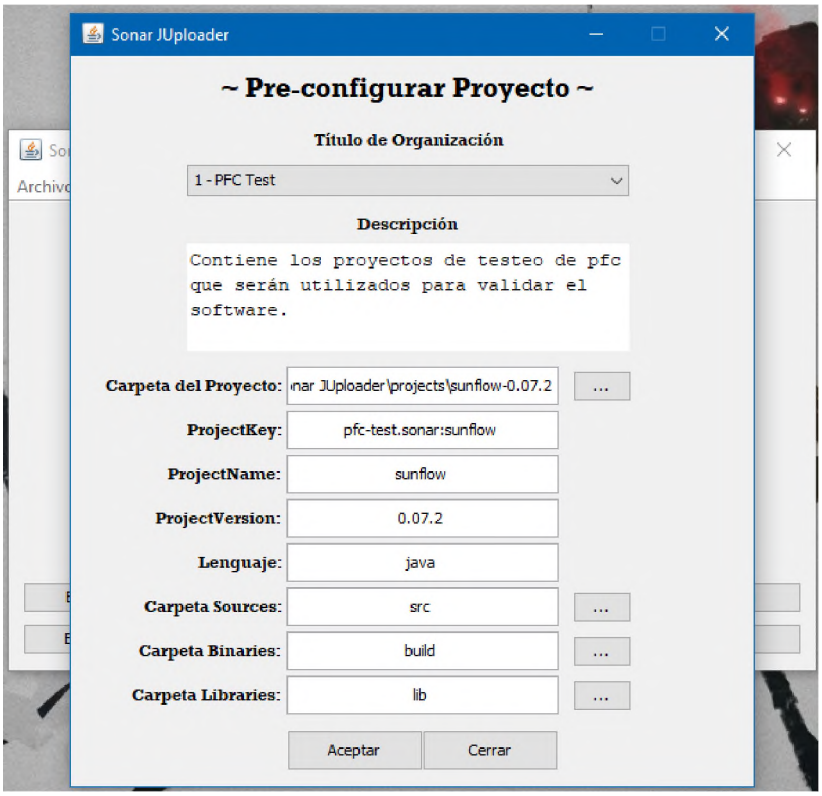

Fig. 2. Interfaz de pre-configuración de proyectos de Sonar JUploader.

Se introduce la versión del proyecto, el lenguaje de programación del mismo, y las ubicaciones de los ficheros de código fuente (sources, en java son ficheros .java), los ficheros compilados (binaries, en java son ficheros .class) y las librerías utilizadas por el mismo (libraries, en java son ficheros .jar).

De forma alternativa, en caso de no querer especificar dichos directorios, predeterminadamente se carga automáticamente el carácter ".", de manera que la herramienta Sonar Scanner tratará la carpeta del proyecto como base para todos los ficheros.

Una vez cargada toda la información requerida, al presionar el botón "Aceptar", se generará el fichero de configuración "sonar-project.properties" respectivo, y dicho proyecto estará preparado para ser analizado.

3.4 Análisis de proyectos

La funcionalidad principal de la aplicación es la ejecución de análisis de proyectos, donde se automatizó el proceso de llamado del análisis; evitando la necesidad de usar la consola de comandos y permitiendo analizar cualquier número de proyectos sin más intervención, en lugar de sólo uno.

6

Como se visualiza en la Fig. 3, se puede elegir entre las organizaciones cargadas del usuario, y obtener automáticamente los datos para realizar el análisis. Esto se omite si se ingresa como invitado a la aplicación, debiendo ser suministrados manualmente al no contar con organizaciones cargadas.

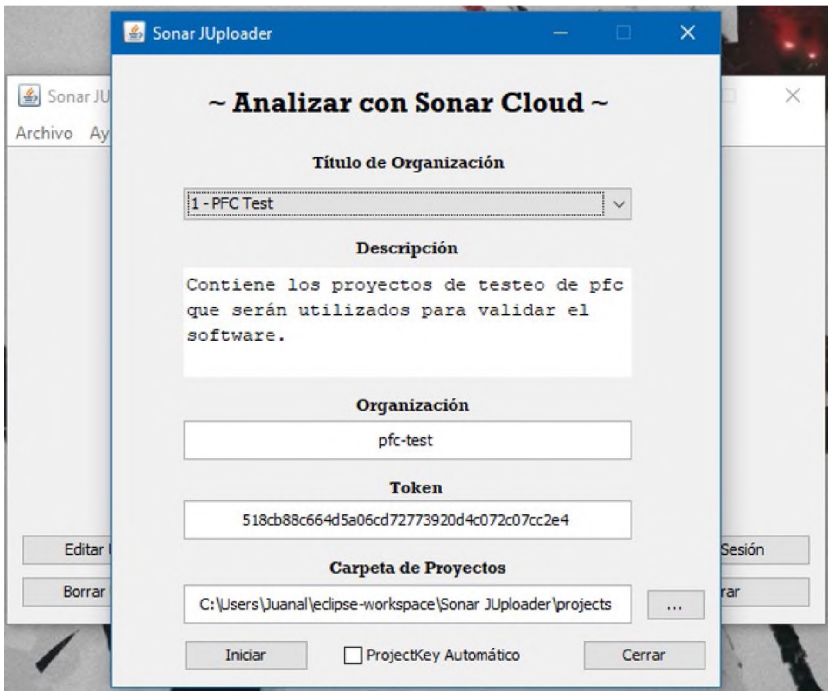

Fig. 3. Interfaz de análisis de proyectos de Sonar JUploader.

Aun así, todos los campos son editables, en caso de requerirse la alteración de los valores para testeos, o análisis en diferentes directorios, en caso de que estos varíen de forma regular.

Al presionar el botón "Iniciar", comenzará el análisis de todos los proyectos dentro del directorio, uno detrás de otro, como se mostró anteriormente, siendo luego subidos a la plataforma de Sonar Cloud donde podrán observarse los resultados. Puede marcarse la opción de "ProjectKey Automático" para casos en los que se analicen proyectos con una organización diferente a la que figura en el fichero de configuración sonar.properties, para evitar el trabajo de editarlos individualmente.

3.5 Visualización de reportes

Es posible consultar información acerca de los análisis ejecutados, por ejemplo, qué organización, fecha y hora, cuantos proyectos analizó, cuáles y la dirección de la carpeta en la cual se ubicaban, como puede observarse en la Fig. 4.

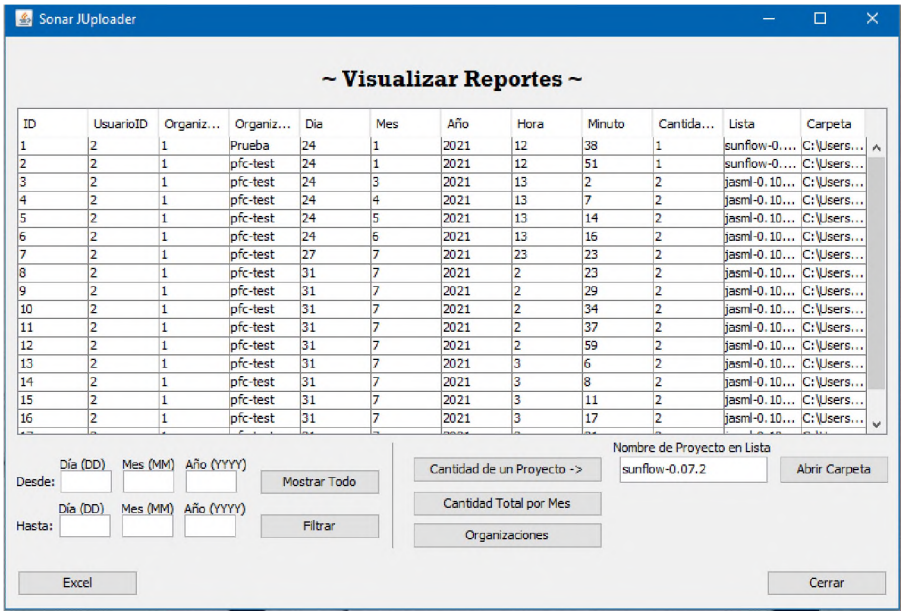

Fig. 4. Interfaz de visualización de reportes de Sonar JUploader.

Además, pueden filtrarse los datos para mostrar en la tabla los análisis realizados entre dos fechas especificadas, los registros mostrados pueden ser exportados a un fichero de Excel, y si se selecciona una fila correspondiente a un análisis, puede usarse el botón "Abrir Carpeta" para automáticamente dirigirnos al directorio utilizando el explorador de archivos.

El botón "Cantidad de un Proyecto" despliega un gráfico de barras con los análisis ejecutados de un proyecto específico mensualmente, y la cantidad total de análisis realizados al mismo como se visualiza en la Fig. 5.

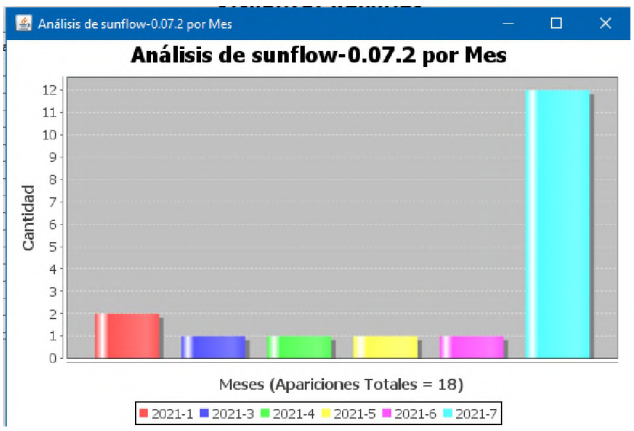

Fig. 5. Reporte de cantidad de un proyecto de Sonar JUploader.

De manera similar, los botones “Cantidad Total por Mes” y “Organizaciones” despliegan gráficos de barras del total de análisis ejecutados de un proyecto y total realizados por organización mensualmente.

4 Validación

Para validar la herramienta se crearon dos organizaciones en Sonar Cloud, Corpus-2021 [19] y PFC-Test [20], donde se registraron los resultados del análisis de una colección de proyectos software basada en el Qualitas Corpus [9]. Corpus-2021 [21] es la versión actualizada del Qualitas Corpus y contiene las versiones estables más recientes de cada proyecto dentro de la colección original. Esta colección de 110 proyectos codificados en java fue analizada, primero con Sonar Scanner y segundo utilizando Sonar Juploader; y los resultados fueron guardados en Corpus-2021 y PFC-Test respectivamente. Una vez finalizados los procesos de análisis, se calcularon los tiempos registrados en Sonar Cloud y fueron comparados para determinar la efectividad de la herramienta construida.

En la Fig. 6 se puede observar el listado de estos proyectos dentro de la organización de Sonar Cloud “Corpus-2021” [19], donde se encuentran algunos resultados de métricas evaluadas.

El primer análisis del lote de proyectos utilizando Sonar Scanner inició el 26 de mayo del 2021 a las 10:27 PM, y finalizó el 1 de junio del 2021 a las 4:16 AM; tomando así un total de 125 horas y 49 minutos, es decir, 69.26 minutos por proyecto. Es necesario aclarar que se descartó el tiempo del primer proyecto (Ant) porque el mismo fue analizado con mucha anterioridad (12 de marzo del 2021), y su inclusión hubiese distorsionado el total de tiempo transcurrido.

El segundo utilizando la herramienta Sonar JUploader, inició el 24 de agosto del 2021 a las 4:00 AM, y finalizó ese mismo día a las 12:28 PM; tomando así un total de 8 horas y 28 minutos. El resultado obtenido indicó que se habían analizado correctamente 106 de los 110 proyectos, habiendo incurrido en fallo solamente 4 de ellos, debido a la pérdida momentánea de la conexión de internet requerida para la sincronización con el servidor de Sonar Cloud. Se realizó luego un segundo análisis con los 4 proyectos faltantes, iniciando también el 24 de agosto del 2021 a las 8:21 PM, y finalizando ese mismo día a las 9:23 PM; totalizando una hora y 2 minutos. Sumando estos dos análisis, los 110 proyectos fueron analizados en 9 horas y 30 minutos, es decir, 5.18 minutos por proyecto.

Como se puede evidenciar se pudo reducir el tiempo promedio de análisis por proyecto de 69.26 minutos a 5.18 empleando Sonar Juploader, presentando una reducción de 64.08 minutos por proyecto. Además cabe remarcar que, en este último, el proceso no fue supervisado hasta su finalización, en cambio el primer análisis requirió un trabajo manual ininterrumpido. Se determina así que la herramienta es capaz de agilizar los tiempos necesarios para generar métricas de la plataforma Sonar Cloud de una gran cantidad de proyectos de forma automatizada.

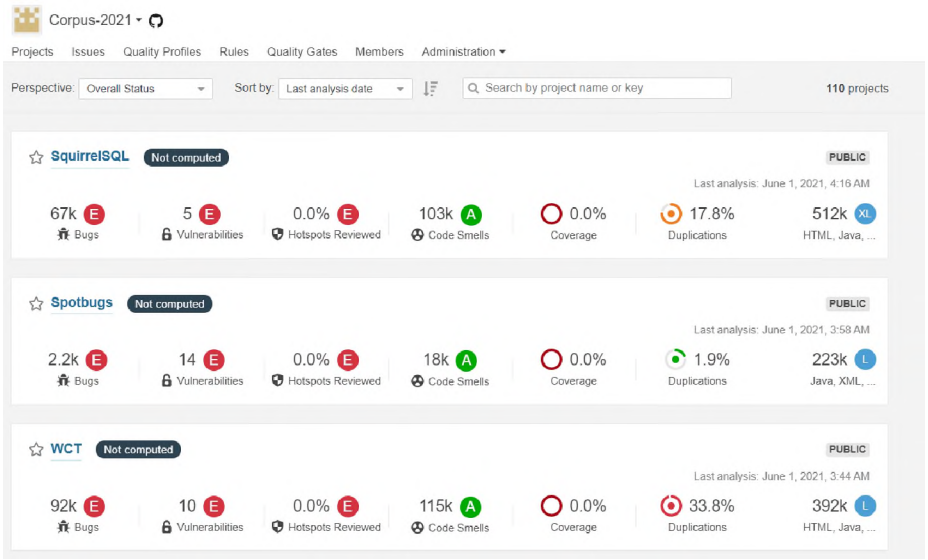

Fig. 6. Proyectos analizados visualizados en la organización corpus-2021.

5 Conclusión y futuros trabajos

Este artículo presenta Sonar Juploader, herramienta para el soporte en el cómputo de métricas de código fuente dirigida a los grupos de investigación en la calidad de software. Su funcionalidad principal consiste en el análisis de un lote de proyectos de Sonar Cloud de manera automatizada, y además permite administrar organizaciones de Sonar Cloud; pre-configurar proyectos rápidamente para su análisis; y visualizar reportes en base a los análisis realizados a lo largo del tiempo. La solución fue validada con una colección de 110 proyectos Java, demostrando que es capaz de disminuir el tiempo de análisis un 92.52% comparando con el análisis manual de Sonar Scanner.

En el futuro consideramos integrar Sonar Juploader con otras herramientas para el análisis estático del código fuente para ampliar el número de métricas generadas, como por ejemplo CKJM, que calcula métricas orientadas no disponibles en Sonar Cloud como cantidad de hijos, profundidad de árbol de herencia, acoplamiento entre objetos, entre otras. Otra alternativa podría ser desarrollar una nueva aplicación que integre el entorno de análisis de CKJM.

Referencias

1. Parnas, D. L. Some software engineering principles. in Software fundamentals: collected papers by David L. Parnas 257–266 (2001).
2. Irrazabal, E., Vásquez, F., Díaz, R. & Garzás, J. Applying ISO/IEC 12207:2008 with SCRUM and Agile Methods. Commun. Comput. Inf. Sci. 155 CCIS, 169–180 (2011).

10

3. Lehman, M. M. Laws of software evolution revisited. in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) vol. 1149 108–124 (Springer Verlag, 1996).
4. Kitchenham, B. & Pfleeger, S. L. Software quality: the elusive target. IEEE Softw. 13, 12–21 (1996).
5. Garvin, D. A. What Does "Product Quality" Really Mean? MIT Sloan Management Review 25–43 (1984).
6. Vidal, S. A., Bergel, A., Marcos, C. & Díaz-Pace, J. A. Understanding and addressing exhibitionism in Java empirical research about method accessibility. Empir. Softw. Eng. 21, 483–516 (2016).
7. Vidal, S., Bergel, A., Díaz-Pace, J. A. & Marcos, C. Over-exposed classes in Java: An empirical study. Comput. Lang. Syst. Struct. 46, 1–19 (2016).
8. Vázquez, H. C., Bergel, A., Vidal, S., Díaz Pace, J. A. & Marcos, C. Slimming javascript applications: An approach for removing unused functions from javascript libraries. Inf. Softw. Technol. 107, 18–29 (2019).
9. Tempero, E. et al. The Qualitas Corpus: A curated collection of Java code for empirical studies. in Proceedings - Asia-Pacific Software Engineering Conference, APSEC 336–345 (2010). doi:10.1109/APSEC.2010.46.
10. Ceccato, M., Capiluppi, A., Falcarin, P. & Boldyreff, C. A large study on the effect of code obfuscation on the quality of java code. Empir. Softw. Eng. 20, 1486–1524 (2015).
11. Okutan, A. & Yıldız, O. T. Software defect prediction using Bayesian networks. Empir. Softw. Eng. 19, 154–181 (2014).
12. Behnamghader, P. et al. A scalable and efficient approach for compiling and analyzing commit history. in International Symposium on Empirical Software Engineering and Measurement (IEEE Computer Society, 2018). doi:10.1145/3239235.3239237.
13. Carruthers, J. A., Diaz Pace, J. A. & Irrazabal, E. A. A Systematic Mapping Study of Empirical Studies performed with Collections of Software Projects. (2021) https://arxiv.org/abs/2109.07624.
14. CKJM - Chidamber and Kemerer Java Metrics, https://www.spinellis.gr/sw/ckjm, último acceso 16/09/2021.
15. PMD Source Code Analyzer, https://pmd.github.io, último acceso 16/09/2021.
16. Spotbugs, https://spotbugs.github.io, último acceso 16/09/2021.
17. Sonar Cloud Documentation, https://sonarcloud.io/documentation, último acceso 16/09/2021.
18. Sonar Cloud Sonar Scanner Documentation, https://sonarcloud.io/documentation, último acceso 16/09/2021.
19. Corpus-2021 Sonar Cloud Organization, https://sonarcloud.io/organizations/corpus-2021, último acceso 16/09/2021.
20. PFC-Test Sonar Cloud Organization, https://sonarcloud.io/organizations/pfc-test, último acceso 16/09/2021.
21. Corpus-2021 Github Organization, https://github.com/Corpus-2021, último acceso 16/09/2021.