Ciencias de la Información Vol. 46, No. 1, enero-abril, pp. 47 - 52, 2015

Nuevos plugins para la herramienta Knime para el uso de sus flujos de trabajo desde otras aplicaciones

Ariel Izquierdo Iglesias
Lisandra Bravo Ilisástigui
Taymí Ceruto Cordovéz
Diana Martin Rodríguez

Konstanz Information Miner (KNIME) es una de las herramientas de minería de datos más populares en el mundo. Esta herramienta orienta de manera fácil el proceso de descubrimiento de conocimiento en los datos a través del diseño de un flujo de trabajo. Sin embargo, KNIME no contiene las funcionalidades que permitan utilizar los modelos de minería de datos obtenidos por sus flujos de trabajo desde otras aplicaciones, sin pagar ninguna licencia por ello. El uso de estos modelos les permitiría a los usuarios de otras aplicaciones utilizar este conocimiento sin tener que desarrollar todo el proceso de descubrimiento de conocimiento en los datos. En este trabajo se propone IntegrationServices.KNIME, representa un conjunto de nuevos plugins para la herramienta KNIME para usar los flujos de trabajo de esta herramienta desde otras aplicaciones. Estos plugins *implementan dos servicios de integración que permiten exponer funcionalidades de KNIME como servicios para ser usados por otras aplicaciones. Los resultados obtenidos al evaluar la calidad de los servicios propuestos muestran su efectividad.*

Palabras clave: KNIME, flujos de trabajo, servicios de integración, minería de datos.

RESUMEN

ABSTRACT

Konstanz Information Miner (KNIME) is one of the most popular data mining tools in the world. This tool guides the process of knowledge data discovery through the design of workûows in an intuitive way. However, KNIME does not have functionalities to use the data mining models obtained by its workflow from other applications, without pay any license for that. The uses of these models for the users of other applications would allow the use of the knowledge obtained without develop all the process of knowledge data discovery. In this paper, we propose IntegrationServices.KNIME, a set of new plugins for KNIME to use its workflow from other applications. These plugins implements two integration services, which presents the functionalities of KNIME as services to be used by other applications. The effectiveness of the services proposed is validated through different quality tests.

Keywords: KNIME, workflow, integration services, knowledge data.

Introducción

El aumento del volumen de información almacenada en diversas fuentes de datos se ha incrementado exponencialmente en las últimas décadas. Este notable crecimiento ha propiciado el desarrollo de nuevas técnicas, como la minería de datos (MD), que permitan realizar diferentes análisis de la información para apoyar la toma de decisiones. Varios autores (Hand, Mannila et al. 2001; Hernández Orallo, Ramírez Quintana et al. 2004) coinciden en que la

MD pretende extraer el conocimiento de mas interes a partir de conjuntos de datos grandes y complejos. La MD es una de las fases del «Proceso de Descubrimiento de Conocimientos en los Datos» (Hernández Orallo, Ramírez Quintana et al. 2004).

En los últimos años se han desarrollado diferentes herramientas para facilitar el descubrimiento de conocimiento en grandes en grandes volúmenes de información, vinculados por ejemplo a la ingeniería, la ciencia o los negocios. Dentro de estas herramientas se destaca la herramienta KonstanzInformationMiner (KNIME) (Berthold, Cebron et al. 2007), desarrollada originalmente en el departamento de bioinformática y minería de datos de la Universidad de Constanza, Alemania. KNIME ha demostrado ser muy útil debido a que es una herramienta de código abierto, multiplataforma, y compuesta por módulos ensamblables que permiten extenderla de forma muy fácil. Además ofrece una gran variedad de funcionalidades extras al flujo básico de la MD y se adapta al uso empresarial.

La herramienta KNIME guía el desarrollo del proceso de descubrimiento en los datos a través del diseño de un flujo de trabajo, en el cual se interconectan varios nodos entre sí para desarrollar todas las fases que lo componen. El resultado de ejecutar este flujo cuando se aplica alguna técnica de MD es representado por patrones y modelos de MD, por ejemplo: reglas de asociación, árboles de decisión, grupos, entre otros (Han and Kamber 2006).

El uso de los modelos/patrones de MD generados por KNIME resulta muy útil para aplicaciones que no se dediquen a guiar y desarrollar el proceso de descubrimiento de conocimiento en los datos. Debido a que el acceso a estos modelos le permitirá a los usuarios de estas aplicaciones utilizar el conocimiento obtenido para apoyar la toma de decisiones, sin tener que asumir el desarrollar desde un inicio todo el proceso de descubrimiento de conocimiento. Sin embargo, KNIME en su versión de escritorio (desktop) no ofrece una solución que permita utilizar sus flujos de trabajo desde otras aplicaciones.

En el año 2011 fue publicada KNIME SERVER (AG 2011) como solución al problema de integrar KNIME con otras aplicaciones. KNIME SERVER ofrece las mismas funcionalidades que KNIME en su versión escritorio, pero también permite utilizar los flujos de trabajo desarrollados en KNIME desde otras aplicaciones a través de servicios web (V. Caselli 2008), además de muchas otras funcionalidades. Sin embargo, KNIME SERVER es una herramienta propietaria y el costo de utilizarla implica el pago de su licencia por varios miles de euros anuales, por lo que no puede ser utilizada por organizaciones que no cuentan con el presupuesto para pagarla.

En este trabajo proponemos un conjunto de nuevos *plugins* para la herramienta KNIME, llamados integrationServices. KNIME para usar los flujos de trabajo de esta herramienta desde otras aplicaciones, sin costo alguno para los usuarios. Para este objetivo, los *plugins* implementan dos servicios de integración (Rosen, Lublinsky et al. 2008), los cuales permiten exponer como servicios a ser usados por otras aplicaciones, las funcionalidades de KNIME . Estos servicios permitirán a los usuarios de otras aplicaciones cargar y ejecutar los flujos de trabajo, así como obtener el modelo de MD que haya sido seleccionado. Además, permitirán configurar los parámetros de los flujos de trabajo, por medio de sus variables de flujo y sus credenciales.

Materiales y Métodos

Konstanz Information Miner

KNIME (Berthold, Cebron et al. 2007) es una plataforma de MD libre que permite el desarrollo de modelos de MD. Esta herramienta direcciona el proceso de descubrimiento de conocimiento en los datos en todas sus fases, a través de un ambiente de trabajo visual e interactivo, lo que le permite al usuario ir conformando paso a paso el flujo de descubrimiento de conocimiento de una manera intuitiva y fácil Además KNIME ofrece múltiples funcionalidades que facilitan el desarrollo de patrones y modelos de MD. Los autores de (Berthold, Cebron et al. 2007) plantean que KNIME se ha diseñado basándose en el principio de la modularidad. Esta propiedad le permite a KNIME ser extendida muy fácilmente con la adición de nuevos *plugins* que contendrán funcionalidades nuevas. KNIME se desarrolló inicialmente por el departamento de bioinformática y MD de la Universidad de Constanza, Alemania. Actualmente la organización KNIME.com GmbH, radicada en Zúrich, Suiza, continúa su desarrollo y además presta servicios de formación y consultoría. En (Meinl, Cebron et al. 2008) se plantea que KNIME fue diseñado basado en tres principios fundamentales:

• Ambiente de trabajo visual e interactivo: Los flujos de trabajo se forman arrastrando los elementos al área de trabajo, de forma que sea fácil e intuitivo para el usuario.

• Modularidad: Las unidades contenedoras de datos o de procesamiento o no deben depender unas de las otras. De esta forma se puede distribuir la computación de cada unidad de la forma que se desee. Además permite la implementación de algoritmos de forma independiente. No hay tipos de datos predefinidos, por lo que se pueden definir nuevos tipos con especificaciones propias. Los nuevos tipos de datos pueden declararse compatibles con otros existentes.

• Extensibilidad de forma sencilla: Adición de nuevas unidades de procesamiento, visualización y tratamiento de datos, debe ser una tarea fácil de realizar, evitando complicar el proceso con procedimientos engorrosos de instalación/desinstalación.

KNIME fue concebida como herramienta gráfica, en la cual se construyen flujos de trabajo que van guíando el proceso de descubrimiento de conocimiento en los datos. Estos flujos se componen de nodos y flechas que se despliegan y combinan de manera gráfica e interactiva. Los nodos son las unidades básicas que procesan datos y/o un modelo, tienen puertos de entrada y salida, que son los elementos a los que se engarzan las flechas. Las flechas transportan datos o modelos, que son resultado del trabajo realizado por algún nodo, además indican el orden de ejecución y flujo de la información (Meinl, Cebron et al. 2008).

Un flujo por lo general comienza con un nodo que carga los datos desde alguna fuente de datos, que suelen ser archivos de texto o base de datos. Los datos importados se almacenan en una tabla interna que se compone de columnas con un determinado tipo de datos y un número arbitrario de filas que se ajusten a las especificaciones de la columna (Meinl, Cebron et al. 2008). Estas tablas de datos se envían a lo largo de las

conexiones a otros nodos que se ocupan de las restantes fases del proceso de descubrimiento de conocimiento en los datos. Por lo general luego de estos nodos de carga se encuentran nodos que se encargan de la etapa de preparación de los datos.

Como próximo paso se colocan los nodos que utilizan algoritmos de MD o de aprendizaje automático para obtener modelos predictivos y descriptivos. Para ver los resultados de un flujo de trabajo están disponibles varios nodos de visualización, que muestran los datos o los modelos de varias formas (Meinl, Cebron et al. 2008). La Figura 1 muestra un ejemplo de un flujo de trabajo con nodos de acceso a datos, de preparación de los datos y de algoritmos de MD.

KNIME se puede extender fácilmente porque está basada en la Plataforma de Cliente Enriquecido de Eclipse (Eclipse RCP, por sus siglas en inglés) (Vogel 2013). Esta plataforma permite que las aplicaciones sean extensibles al tener una arquitectura modular basada en módulos o *plugins*, utilizando el contenedor de *plugins* de Eclipse Equinox como implementación del conjunto de especificaciones OpenService Gateway Iniciative (OSGi) (Alliance 2013). El uso de las especificaciones OSGi permite el desarrollo de *plugins* y servicios, a partir de los cuales se construyen aplicaciones modulares y extensibles.

Una de las mayores ventajas que ofrece a KNIME esta característica es su fácil extensibilidad, de manera que los usuarios pueden añadir nuevas funcionalidades de acuerdo a sus necesidades. KNIME puede ser extendido a través de estos *plugins*, la mayoría de los cuales son nuevos nodos, aunque las extensiones pueden ser a cualquier parte de la arquitectura. Existe una amplia comunidad de usuarios y desarrolladores que tienen su propio sitio de descargas para extensiones desarrolladas por esta herramienta. En la actualidad, varios investigadores han desarrollado extensiones para procesamiento de datos químicos y biológicos, para el tratamiento de ficheros XML, procesamiento de imágenes, ficheros de comandos de Matlab, entre otras.

Resultados y discusión

IntegrationS ervices.KNIME: nuevos *plugins* para el uso de LOS flujos de trabajo DE knime deSDE otras aplicaciones

Para el empleo de los flujos de trabajo de KNIME desde otras aplicaciones se emplea el diseño de servicios de integración (Rosen, Lublinsky et al. 2008) implementados en nuevos *plugins* para la herramienta KNIME, a los cuales hemos llamado integration Services.KNIME. Se han usado servicios de integración porque estos exponen las funcionalidades de aplicaciones como los servicios para ser usados por el resto de la organización o por otras aplicaciones.

Primeramente, debemos tener en cuenta que los usuarios de las otras aplicaciones pudieran acceder a la información de los modelos de MD que ofrecen estos flujos y también pudieran configurar sus parámetros para ajustar el conocimiento extraído según sus necesidades. Los usuarios de estas aplicaciones solo necesitarían realizar las siguientes actividades para obtener el modelo de MD:

1. Seleccionar el modelo a obtener.
2. Introducir usuario y contraseña de la BD de ser necesario.
3. Configurar los parámetros de entrada del modelo.
4. Ejecutar el modelo de MD.

Para diseñar los servicios que permitirán a usuarios de otras aplicaciones desarrollar las actividades que se han comentado anteriormente, es importante destacar las siguientes características de los flujos de trabajo de KNIME:

• Los modelos de MD son construidos en nodos de tipo «learner», por cada modelo construido interviene un nodo de este tipo, por lo que son identificados como los nodos que extraen conocimiento interesante para los usuarios.

• Los flujos también facilitan a los usuarios el poder autenticarse a una BD (fuente de datos del flujo) para poder ejecutar el flujo de trabajo. Las variables de flujo y las credenciales de flujo conforman la configuración del flujo de trabajo. Ambas propiedades permiten readaptar el flujo antes de su ejecución.

• Los flujos en KNIME permiten a través de variables de flujo, que los usuarios cambien los parámetros en la configuración de los nodos. Esta funcionalidad permite una mayor flexibilidad a los flujos de cara al usuario, que tendrá la posibilidad de adaptar el flujo a su conveniencia. De no contar con variables de flujo, el usuario se limitaría a ejecutar el flujo con la configuración fija en cada nodo. De esta manera se brinda al usuario la posibilidad de escoger los parámetros de entrada para la ejecución.

A partir de la situación comentada se definen los siguientes requerimientos funcionales que deben tener en cuenta los servicios que serán diseñados:

1. Funcionalidad: Listar los nodos de tipo «learner».
   Descripción: A partir de este requisito se le informará al usuario de otra aplicación del listado de modelos de MD que puede obtener, es decir, el conocimiento que puede extraer de su problema.

2. Funcionalidad: Mostrar las variables de flujo.
   Descripción: Los servicios deben permitir listar las variables de flujo, las cuales permiten que los usuarios cambien los parámetros en la configuración de los nodos. La información que se maneja son el nombre, el tipo y el valor de la variable de flujo.

3. Funcionalidad: Modificar las variables de flujo.
   Descripción: A partir de que el usuario introduzca las credenciales y seleccione

Figura 1: Ejemplo de flujo de trabajo de KNIME para la tarea de Clasificación

los valores de las variables de flujo, los servicios deben permitir que se actualicen las variables de flujo (Solo se permiten actualizar los tipos y los valores de las variables).

4. Funcionalidad: Mostrar las credenciales de flujo.
   Descripción: Los servicios deben permitir listar las credenciales de flujo y esta información que se maneja son el nombre de la credencial, el usuario y la contraseña.

5. Funcionalidad: Actualizar las credenciales de flujo.
   Descripción: A partir de que el usuario introduzca las credenciales y seleccione los valores de las variables de flujo, los servicios deben permitir que se actualicen las credenciales de flujo (Solo se permiten actualizar los usuarios y las contraseñas).

6. Funcionalidad: Obtener el modelo de MD.
   Descripción: Los servicios deben ser capaces de ejecutar el flujo de trabajo con la configuración deseada. Una vez ejecutado el flujo, se debe obtener como resultado final el modelo de MD, que será un documento en formato PMML.

7. Funcionalidad: Obtener la configuración del flujo de trabajo.
   Descripción: Los servicios deben permitir crear un documento XML con la configuración deseada por el usuario. Este documento será el usado para crear el fichero en formato XML que almacenará la configuración.

A partir de las funcionalidades antes descritas, se proponen dos servicios de integración: GestionFlujosTrabajo y ConfiguracionFlujoTrabajo. El servicio GestionFlujosTrabajo se ocupa de la obtención de los modelos de MD, mientras que el servicio ConfiguracionFlujoTrabajo agrupa las operaciones necesarias para configurar los parámetros de los flujos de trabajo. A continuación se describen la definición de cada uno de estos servicios, especificando sus operaciones y el diseño de sus interfaces.

El servicio GestionFlujosTrabajo realiza las siguientes operaciones:

1. obtenerIdFlujos: obtiene el listado de los indicadores de los flujos de trabajo que están implementados en el workspace .

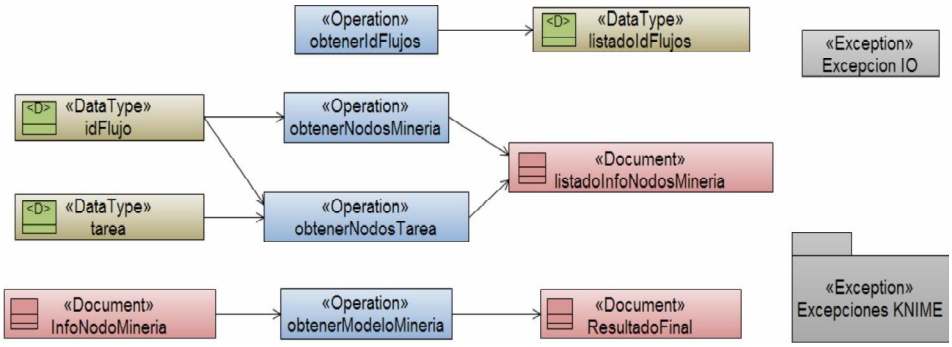

Figura 1: Diagrama de definición utilizado para modelar la interfaz del servicio GestionFlujosTrabajo

de KNIME

2. obtenerNodosMineria: obtiene todos los nodos tipo «learner» a partir del identificador de un flujo de trabajo.

3. obtenerNodosTarea: obtiene los nodos de tipo «learner» a partir de una tarea de MD especificada y del identificador de un flujo de trabajo.

4. obtenerModeloMineria: obtiene el modelo de minería de datos a partir de especificar un nodo de tipo «learner» y el identificador del flujo al que pertenece.

El servicio ConfiguracionFlujoTrabajo realiza las siguientes operaciones:

1. obtenerVariablesFlujo: obtiene las variables de flujo a partir del identificador de un flujo de trabajo.

2. obtenerCredencialesFlujo: obtiene las credenciales de un flujo a partir de especificar su identificador.

3. adicionarVariablesFlujo: permite adicionar nuevas variables de flujo.

4. actualizarCredencialesFlujo: permitir que se actualicen las credenciales del flujo de trabajo.

5. obtenerConfiguracionFlujo: permite obtener la configuración del flujo que desea por el usuario.

El diseño de las interfaces de servicio describe las relaciones entre las operaciones y sus parámetros (Rosen, Lublinsky et al. 2008). Para representar de manera visual la interfaz de un servicio se utiliza el diagrama de definición de servicio. En las Figura 2 y 3 se representa respectivamente el diagrama de definición del servicio:
- GestionFlujosTrabajo y
- ConfiguracionFlujoTrabajo

Los diagramas mostrados en las figuras 2 y 3 están compuestos por cuatro columnas. La primera columna muestra los parámetros de entrada de cada operación, la segunda columna describe las operaciones del servicio, la tercera columna presenta los parámetros de salida y por último la cuarta

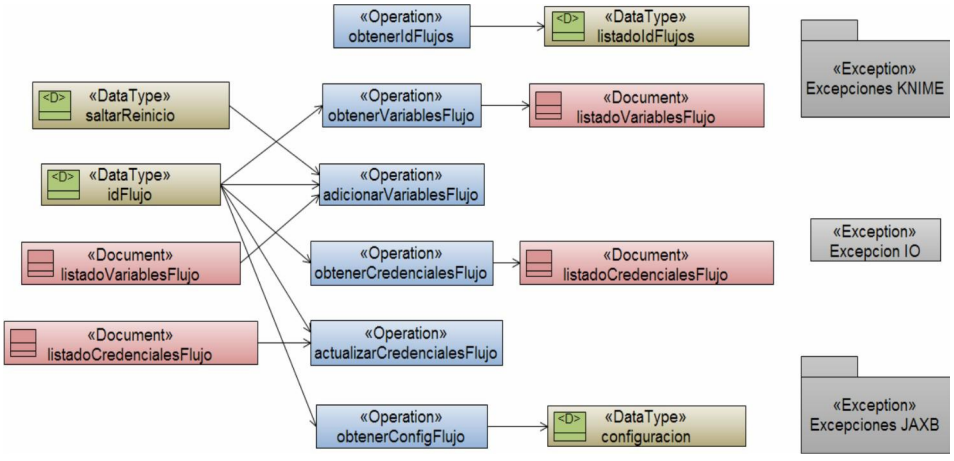

Figura 1: Diagrama de definición utilizado para modelar la interfaz del servicio

las excepciones que pueden devolverse.

Como se ha comentado en la sección 2, KNIME posee una arquitectura modular basada en *plugins* y orientada a servicios, que le permite ser extendida mediante la adición de nuevos *plugins* a cualquier parte de la arquitectura. Por esta razón, se han implementado estos servicios en los *plugins*integrationServices.KNIME, encapsulando las funcionalidades necesarias para el uso de los flujos de trabajo desde otras aplicaciones.

Según se plantea en (Vogel 2013), resulta una buena práctica de diseño dividir la implementación y la interfaz del servicio en *plugins* distintos. Esta práctica presenta como ventaja que se puede contar con varias versiones de implementación para una misma interfaz de servicio. A continuación se describen cada uno de los *plugins* de integrationServices.KNIME añadidos a la herramienta:

1. IntegrationServices.KNIME.IGestion FlujosTrabajo: *plugin* que contiene la interfaz del servicio GestionFlujosTrabajo y se encarga de exponer sus operaciones a los consumidores del servicio.

2. IntegrationServices.KNIME.ImGestion FlujosTrabajo: *plugin* que contiene la implementación del servicio GestionFlujos Trabajo. Este *plugin* realiza la interacción con el *plugin integrationServices.KNIME. lSFlujosTrabajo*, que es la responsable de la ejecución de las operaciones internas del servicio.

3. IntegrationServices.KNIME.IConfiguracion FlujoTrabajo: *plugin* que contiene la interfaz del servicio ConfiguracionFlujoTrabajo y se encarga de exponer sus operaciones a los consumidores del servicio.

4. integrationServices.KNIME.ImConfiguracion FlujoTrabajo: *este plugin* que contiene implementación del servicio Configuracion FlujoTrabajo. Este *plugin* realiza la interacción con el *plugin*integration *Services. KNIME. lSFlujosTrabajo*, que es la responsable de la ejecución de las operaciones internas del servicio.

5. integrationServices.KNIME.lS Flujos Trabajo: Este *plugin* es la base de la implementación de los servicios y se encarga de implementar el flujo de procesamiento interno de las operaciones de los servicio. Este *plugin* transforma las entidades que dispone KNIME para el uso de los flujos de trabajo en información que sale en forma de documentos a los consumidores de los servicios.

Por último, destacar que para lograr exportar los servicios diseñados fuera del contenedor de *plugins* de Eclipse se utilizó el framework Apache CXF-DOSGi (disponible en: http://cxf.apache.org/distributed-osgi.html), agregando sus *plugins* a KNIME.

Resultado de las pruebas para medir la calidad de los servicios

Para validar y verificar las características principales de la calidad de los servicios se realizaron un conjunto de pruebas. El propósito fundamental de estas pruebas es determinar el rendimiento del servidor ante el aumento de cargas simuladas. Las cargas en este caso son el número esperado de usuarios concurrentes que solicitan los servicios y que realizan un número de transacciones durante el tiempo que dura la carga.

Para la realización de las pruebas de rendimiento de los servicios se utilizó la herramienta Apache JMeter (disponible en: http://jmeter.apache.org/). Esta herramienta de carga fue diseñada para realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones de tipo cliente / servidor.

Por otra parte, en el resultado de las pruebas también influyen las propiedades físicas del servidor web utilizado, en el cual el tipo de CPU es Dual-Core E7300, 2133 MHz; la memoria física es de 4GB DDR2, 800 MHz; el espacio físico es de 1TB y el chipset de la tarjeta madre es nVIDIA nForce 780i SLI.

El resultado de las pruebas muestra la forma en que el rendimiento varía con la carga, mostrando como el número de usuarios modifica el tiempo de respuesta de los servicios. Para facilitar la interpretación de las pruebas hemos utilizado tres escenarios diferentes, donde se modifican el número de usuarios que acceden a los servicios (10, 25 y 200), el total de peticiones a las operaciones de los dos servicios (8000, 20000, 16000) y el tiempo entre usuarios (1; 0.4; 0.25). La Tabla 1 muestra los resultados obtenidos para los tres escenarios, mostrándose el rendimiento del servidor, el porciento de peticiones no satisfactorias, el tiempo medio de respuesta y la latencia de los servicios, donde la latencia es el tiempo que tomó prestar el servicio, desde el envío de una solicitud hasta la llegada de la respuesta (Yu 2007).

Al analizar los resultados de la Tabla 1 podemos concluir que el servidor soporta un aproximado de 25 usuarios que concurren con un total de hasta 20000 peticiones, para un tiempo de respuesta promedio de 2043 milisegundos. Para las pruebas realizadas el servidor fue capaz de atender casi 12 solicitudes por segundo de manera estable, con un porciento de solicitudes no satisfactorias por debajo del 0,5 % y una latencia de 13991 milisegundos. El tiempo de respuesta y el rendimiento dependen de la carga de trabajo que tenga el servidor web en ese momento.

Las peticiones pueden realizarse con un tiempo de subida entre los usuarios de 0,4 segundos en adelante. A partir de 25 usuarios realizando peticiones en un intervalo inferior a los 0,4 la latencia aumenta considerablemente. Es importante destacar que los 13991 milisegundos de

Figura 1: Resumen de métricas

Características de los Escenarios			Resultados			
Número de usuarios	**Total de peticiones**	**Tiempo entre usuarios (segundos)**	**Rendimiento Servidor**	**% Peticiones no satisfactorias**	**Tiempo medio de respuesta (ms)**	**Latencia (ms)**
10	8000	1	11,6	0,26	828	4135
25	20000	0,4	11,8	0,34	2043	13991
200	16000	0,25	11,4	0,46	16115	97397

latencia obtenidos para 25 usuarios pueden ser reducidos si se aumenta el tiempo transcurrido entre el lanzamiento de los usuarios, que para las pruebas fue solo de 0,4 segundos.

Conclusiones

En este trabajo hemos propuesto un conjunto de nuevos (Vogel 2013)para la herramienta KNIME que permiten usar sus flujos desde otras aplicaciones, sin costo alguno para los usuarios. Estos *plugins* implementan los servicios de integración propuestos para permitir a los usuarios de otras aplicaciones gestionar los flujos de trabajo desarrollados en KNIME. El servicio de integración Gestion FlujosTrabajo a través de las operaciones que ofrece permite cargar y ejecutar los flujos de trabajo, así como obtener el modelo de MD correspondiente. El servicio ConfiguracionFlujoTrabajo contiene las operaciones que brindan la posibilidad de configurar los flujos de trabajo desde otras aplicaciones, mediante sus variables de flujo y sus credenciales. Para exportar los servicios a la Web se utilizó el framework Apache CXF-DOSGi.

Los resultados alcanzados durante las pruebas muestran que el servidor soporta aproximadamente 25 usuarios concurrentes, además de atender casi 12 solicitudes por segundo, de manera estable, con un porciento de solicitudes no satisfactorias por debajo del 0,5 % y una latencia de 13991 milisegundos. Los servicios poseen un alto grado de confiabilidad, ya que para los tres escenarios el porciento de fallos se mantuvo estable por debajo del 0,5%.

Referencias

(2014/10/20/19:47:00). «Equinox.» from http://www.eclipse.org/equinox/files/4/equinox.html.

AG, K. c. (2011). KNIME Server - the Heart of a Corporate KNIME Setup Zurich, Switzerland

Alliance, O. (2013). «OSGi Alliance / Specifications / HomePage.» 2013, from http://ww.osgi.org/Specifications/HomePage.

Berthold, M. R., N. Cebron, et al. (2007). KNIME: The Konstanz Information Miner. Studies in Classiûcation, Data Analysis, and Knowledge Organization (GFKL), Springer.

Han, J. and M. Kamber (2006). Data Mining: Concepts and Techniques, Morgan Kaufmann.

Hand, D., H. Mannila, et al. (2001). Principles of Data Mining. England, London, Massachusetts Institute of Technology.

Hernández Orallo, J., M. J. Ramírez Quintana, et al. (2004). Introducción a la Minería de Datos. Madrid, Universidad Politécnica de Valencia, Departamento de Sistemas Informáticos y Computación.

Meinl, T., N. Cebron, et al. (2008). The Konstanz Information Miner 2.0, University of Konstanz.

Rosen, M., B. Lublinsky, et al. (2008). Applied SOA: Service-Oriented Architecture and Design Strategies Indiana, Estados Unidos, Wiley Publishing, Inc.

V. Caselli, C. B. y. B., Malhar (2008). Web Services and SOA, in Service Oriented Architecture with Java. Birmingham, UK, Packt Publishing Ltd.

Vogel, L. (2013). «Eclipse 4 RCP - Tutorial: Building Eclipse RCP applications based on Eclipse 4.» 2014, from http://www.vogella.com/tutorials/EclipseRCP/article.html.

Vogel, L. (2013). «OSGI Services - Tutorial.» 2013, from http://www.vogella.com/tutorials/OSGiServices/article.html.

Yu, W. D. (2007). Modeling the Measurements of QoS Requirements. Web Service Systems.

Recibido: 20 de agosto de 2014.
Aprobado en su forma definitiva:
9 de enero de 2015

Ariel Izquierdo Iglesias
Complejo de Investigación de Tecnologías Integradas (CITI), La Habana, Cuba
Correo-e.: aizquierdo@ceis.cujae.edu.cu

Lisandra Bravo Ilisástigui
Instituto Superior Politécnico José Antonio Echeverría (CUJAE),
La Habana, Cuba.
Correo-e.: lbravo@ceis.cujae.edu.cu

TaymíCeruto Cordovéz
Instituto Superior Politécnico José Antonio Echeverría (CUJAE),
La Habana, Cuba.
Correo-e.: tceruto@ceis.cujae.edu.cu

Diana Martin Rodríguez
Instituto Superior Politécnico José Antonio Echeverría (CUJAE),
La Habana, Cuba.
Correo-e.: dmartin@ceis.cujae.edu.cu