Reusabilidad en el contexto de Desarrollo de Sistemas para Big Data

Agustina Buccella, Juan Luzuriaga, Alejandra Cechich,
Líam Osycka, Facundo Paterno, Matias Pol'la, Marcos Cruz
Rodolfo Martínez, Rafaela Mazalu, Marcelo Moyano
GIISCO Research Group
Departamento de Ingeniería de Sistemas
Universidad Nacional del Comahue
Neuquen, Argentina
agustina.buccella@fi.uncoma.edu.ar

1. Resumen

Actualmente, el crecimiento de sistemas de Big Data (SBD) está obligando a la comunidad de Ingeniería de Software a replantearse un cambio de paradigma en el desarrollo de estos sistemas. Consecuentemente, en este contexto existen varios desafíos para arquitectos y diseñadores, particularmente sobre los requerimientos que una arquitectura para SBDs debe cumplir; entre ellos, considerar las cinco "Vs" (Volumen, Velocidad, Variedad, Variabilidad y Veracidad). En particular, en nuestra investigación nos centramos en una de estas características, *Variabilidad*, que se refiere a la naturaleza evolutiva de los datos.

Nuestro trabajo se enfoca en incorporar Variabilidad en SBDs a través del modelado de elementos reusables de un dominio – sea este de negocios o tecnológico. Esto nos lleva a incorporar información (y sus posibles usos), a modo de línea de productos software. El presente proyecto tiene como fin desarrollar técnicas y herramientas que mejoren la explotación de grandes volúmenes de datos, favoreciendo el desarrollo de ambientes inteligentes que permitan reusabilidad.

Palabras Clave: Reusabilidad - Líneas de Producto de Software - Big Data

2. Contexto

La línea presentada se inserta en el contexto del *Programa: Desarrollo de Software Basado en Reuso - Parte II (04/F009)*. Directora: Dra. Alejandra Cechich, y *SubProyecto: Reuso Orientado a Dominios - Parte II*. Incluído dentro del Programa. Directora: Dra. Agustina Buccella, Codirector: Mg. Juan Manuel Luzuriaga.

3. Introducción

A pesar de que existen mecanismos para manejar el cambio de esquemas y datos en un modelo relacional [2], alcanzar un nivel similar en Big Data es todavía un desafío importante debido a la naturaleza flexible de los esquemas de almacenes NoSQL [6, 7].

En [3], se presentan seis arquitecturas de referencia[1] propuestas actualmente en la literatura para SBDs y se analiza el cumplimiento de las cinco Vs como requerimientos tradicionales. El análisis se hace de manera transversal; es decir, se analiza cada requerimiento por separado y luego sus interacciones con otros requerimientos. Por ejemplo, en el contexto

[1] Una arquitectura de referencia combina el conocimiento sobre arquitecturas en general con la experiencia en requerimientos específicos de una solución en un domunio de problema.

 RedUNCI - UNdeC. ISBN: 978-987-24611-3-3

de SBDs se explotan almacenes NoSQL que guardan datos con bajo acoplamiento y flexibles; sin embargo, no satisfacen requerimientos de Variabilidad y Veracidad, por lo que, en la práctica, las aplicaciones resultan en modelos ad hoc. De ahí la necesidad de contar con arquitecturas de referencia que faciliten el desarrollo concreto, conociendo los componentes y sus relaciones previamente para permitir el análisis de propiedades. Del análisis realizado en [3], se desprende que sólo una propuesta (Bolster, [8]) satisface los requerimientos de Variabilidad por medio de (1) el almacenamiento del esquema de información de los elementos incorporados; (2) la existencia de estadísticas descriptivas para acceder a la evolución de los datos; y (3) el almacenamiento de la información sobre fuentes de datos usando un repositorio de metadatos (MetaData Management system).

Por otra parte, en [5] se presenta el desarrollo de una arquitectura para un SBD en un caso específico, haciendo uso de una arquitectura de referencia que define una familia de sistemas relacionados. En particular, como el dominio es demasiado amplio, la arquitectura de referencia se ve acotada por medio de casos de uso (ej. visualización y análisis de información geoespacial estratégica, análisis inteligente de señales, etc.). Esta arquitectura de referencia sirve como mecanismo para capturar y compartir conocimiento, conteniendo tanto conocimiento del dominio (casos de uso) como conocimiento de la solución (la correspondencia a tecnologías concretas).

Considerando estas propuestas, nuestro trabajo se enfoca en incorporar Variabilidad en SBDs a través del modelado de elementos reusables de un dominio – sea este de negocios o tecnológico. Esto nos lleva a incorporar información (y sus posibles usos), a modo de línea de productos software.

En particular, los desarrollos en las líneas de productos de software (LPS) [10] se centran en identificar similitudes y variabilidades dentro de dominios particulares para ser reutilizados cuando se desarrollan nuevos productos. A su vez, este reuso de dominios puede ser extendido a subdominos, en especial cuando entre ellos existen relaciones o aspectos similares.

En la Figura 1 se muestra una primera aproximación de los elementos que componen nuestra propuesta de una arquitectura de referencia para SBDs basada en reuso. Como puede verse, los aspectos de negocios (dominio), aplicación (software y análisis) y tecnológicos se abordan en niveles separados; siendo transversales aspectos como el uso/reuso de estándares, taxonomías y conocimiento.

Los componentes principales de la arquitectura son:

- *Taxonomía de Dominio y Estándares*: Las taxonomías específicas de dominio permiten clasificar elementos de acuerdo a determinados criterios, relaciones y propiedades [4]. Su principal objetivo es capturar el conocimiento del dominio basándose en divisiones de las entidades acordes a lo que se intenta especificar. Estas entidades pueden ser objetos del dominio, servicios e incluso cualquier otro elemento o conjunto de elementos que se desee clasificar. Por lo tanto, la creación de taxonomías debe contribuir a la definición de un vocabulario común y controlado para todos los participantes. Al mismo tiempo, para garantizar interoperabilidad y luego reuso, las taxonomías deben construirse en base a los estándares existentes. Así en este componente se deben considerar los estándares creados para la ingeniería de software, los definidos para Big Data y aquellos definidos para el dominio en que se este trabajando. Por ejemplo, en el caso de Big Data en los últimos años han surgido una serie de estándares respecto a su arquitectura de referencia, interoperabilidad, terminología, etc. Estos esfuerzos de estandarización han sido llevados a cabo por el comité ISO/IEC JTC 1/SC 42 (Artificial intelligence)[2] y por el grupo de trabajo del NIST (Big Data Public Working Group - NBD-PWG)[3]. A su vez, la información estandarizada del dominio se refiere a los estándares existentes en el dominio que se este analizando.

[2] https://www.iso.org/committee/6794475.html

[3] https://bigdatawg.nist.gov/

 RedUNCI - UNdeC. ISBN: 978-987-24611-3-3

Figura 1. Arquitectura de referencia para SBDs basada en reuso

- *Activos orientados al dominio*: Estos activos agrupan todos aquellos artefactos de software que son creados para el dominio en el que se esta trabajando. Así, además de incluir a los participantes del desarrollo del SBD, como los ingenieros de software, científicos de datos, desarrolladores, usuarios expertos, etc., involucra los requerimientos del proyecto y del dominio, restricciones, modelos (artefactos de análisis y diseño generados) y casos de uso. Es importante resaltar que estos activos deben generarse a partir de las *taxonomías de dominio y estándares* y de los *activos basados en conocimiento*. De esta forma, se deben crear artefactos enfocados en que puedan ser reusados en el mismo dominio e incluso en otros dominios relacionados (artefactos para reuso), y/o que puedan desarrollarse en base a otros artefactos ya creados (artefactos con reuso). A su vez aquí es importante contemplar la variabilidad. Es decir, estos activos deben considerar los aspectos comunes y aquellos variables dentro del dominio. Por ejemplo, como describimos previamente, es importante crear activos flexibles que puedan adaptarse a la evolución de los esquemas y datos fuentes para que puedan seguir siendo útiles en los análisis realizados.

- *Software/Analítica Reusable*: La analítica de datos (data analytics) es un término abarcativo que se encarga gestionar los datos en todo su ciclo de vida. Como es sabido, los datos sin procesar (raw data) por sí mismos no tienen un significado útil, por lo que la analítica de datos se dedica al proceso de extraer y crear información desde estos datos por medio de la recolección, limpieza, organización, almacenamiento, procesamiento, contextualización, análisis y gobernanza de datos. Existen 4 tipos de categorías para la analítica de datos que dependen de los resultados que producen: descriptiva, diagnóstica, predicitiva y presprictiva. Para cada una de estas categorías existen también diversas técnicas o algoritmos de análisis de datos como clasificación, regresión, clustering, visualización, etc.

  En este componente de la arquitectura, se deben definir y diseñar los tipos de análisis necesarios en el domino junto con la forma de realizarlos, es decir el diseño de

los algoritmos. Esto no es una tarea sencilla ya que cada uno de ellos requiere un conjunto de datos de entrada específico, es decir formatos y estructuras específicas requeridas por cada uno. Al mismo tiempo, se deben documentar los procesos de diseño de estos algoritmos de forma que puedan ser reusados en diferentes dominios.

- *Tecnología*: La tecnología disponible para crear sistemas de Big Data es muy variada. Considerando el cumplimento de las 5 V's que se describieron previamente, existen diferentes tecnologías que se abocan en una o varias de ellas. Por ejemplo, para lidiar con los problemas de la Variabilidad, en cuanto a las diferentes fuentes de datos disponibles en sus muy diversos formatos, tenemos un conjunto de herramientas independientes como OpenRefine[4], Optimus[5], o lenguajes con librerías específicas como R[6] o python[7]. También existen varios frameworks especializados en la analítica de los datos como Spark MLib[8] el cual también forma parte del ecosistema Hadoop. Al mismo tiempo conviven los diferentes tipos de repositorios que permiten almacenar la información extraída desde las fuentes de datos y hacerla disponible para su análisis y visualización. Estos repositorios son muy variados, desde aquellos basados en tecnologías NoSQL como MongoDB[9] o CouchDB[10], sistemas de archivos distribuidos como HDFS[11] (también creado como parte de Hadoop) etc. Finalmente también podemos citar aquellas plataformas y frameworks provistos en la nube que surgen principalmente para lidiar con la escalabilidad. Entre los mas conocidos podemos citar Google Cloud[12], AWS[13] y Azure[14].

- *Activos basados en Conocimiento*: Los repositorios para reuso ponen a disposición un amplio rango de activos que los ingenieros de software pueden usar para desarrollar sistemas y así reducir la necesidad de crear nuevamente componentes que provean la misma funcionalidad. Existen diversas alternativas en la elaboración de estos repositorios (basados en componentes, en modelos, etc.). Las funcionalidades asociadas a SBDs son activos potencialmente reusables en el mismo sentido, agregando además la capacidad de reutilización de los datos en sí mismos; lo que hace que los repositorios de experiencias se conviertan en un elemento clave para alcanzar el reuso de activos de dominio. Respondiendo a las cuestiones abiertas planteadas en [9] sobre cómo alcanzar beneficios efectivos al compartir datos, nuestro enfoque intenta identificar objetivos de los posibles usos (ej. interoperabilidad, integración, etc.) así como determinar en qué medida el uso de formatos o estándares facilita el reuso, o cómo distinguir entre datos/usos potencialmente reusables en contexto.

4. Líneas de Investigación y Desarrollo

En investigaciones previas, hemos realizado amplios avances en lo que respecta al área de LPSs definiendo y refinando una metodología de desarrollo a nivel de subdominos. Dentro de la metodología, hemos presentado sus bases y diseñado artefactos que se utilizan en el análisis de dominios y en el análisis organizacional de una LPS [1] y tienen la particularidad de favorecer el reuso basado en una taxonomía de servicios. Es precisamente esta ventaja la que nos permitió luego realizar extensiones hacia otros subdominios. De esta forma hemos podido así avanzar en el desarrollo de múltiples LPSs basadas en la jerarquía de dominios definida.

[4] https://openrefine.org/
[5] https://hi-optimus.com/
[6] https://www.r-project.org/
[7] https://www.python.org/
[8] https://spark.apache.org/mllib/
[9] https://www.mongodb.com/
[10] https://couchdb.apache.org/
[11] https://hadoop.apache.org/
[12] https://cloud.google.com/
[13] https://aws.amazon.com/
[14] https://azure.microsoft.com/

5. Resultados Obtenidos/Esperados

El objetivo principal de la línea de investigación es *desarrollar técnicas y herramientas que mejoren los procesos y técnicas aplicadas a la explotación de grandes volúmenes de datos, favoreciendo el desarrollo de ambientes inteligentes que permitan reusabilidad*.

Previamente, hemos trabajado en el área de Líneas de Productos, donde hemos definido y aplicado nuevos métodos y técnicas para la creación de LPSs con soportes inteligentes dentro del dominio geográfico que contemplan las particularidades de los subdominios incluidos. A su vez, hemos realizado formalizaciones de reglas y patrones para soportar el desarrollo asistido, de manera que sean lo suficientemente generales para ser aplicados en otros subdominios geográficos.

En la presente investigación, se esperan aplicar estos resultados previos para extender e instanciar el modelo presentado en este artículo. Éste se validará en casos de predicción de calidad del agua en golfos de la Patagonia (San Jorge, San Matías) y en ríos (Río Limay, Río Negro), midiendo el grado de extensibilidad y reusabilidad de los modelos.

6. Formación de Recursos Humanos

El proyecto reúne a 13 investigadores, entre los que se cuentan docentes y alumnos de UNComa, y colaboradores. A su vez, cuenta con dos doctores y un magister. Varios de los docentes-investigadores de GIISCo-UNComa han terminado o se encuentran próximos a terminar carreras de postgrado. Uno de ellos se encuentra finalizando su doctorado en el transcurso de este año, en el área Gestión de Variabilidad. A su vez varios de los integrantes se encuentran finalizando sus tesis de grado. Por último, este año seguiremos con la supervisión del trabajo de 2 becarios EVC-CIN.

Referencias

[1] A. Buccella, A. Cechich, M. Arias, M. Pol'la, S. Doldan, and E. Morsan. Towards systematic software reuse of gis: Insights from a case study. *Computers & Geosciences*, 54(0):9 – 20, 2013.

[2] Carlo Curino, Hyun Jin Moon, Alin Deutsch, and Carlo Zaniolo. Automating the database schema evolution process. *International Journal on Very Large Data Bases*, 22(1):73–98, 2013.

[3] Ali Davoudian and Mengchi Liu. Big data systems: A software engineering perspective. *ACM Computing Surveys*, 53(5), 2020.

[4] I. Hunink, E. Rene, S. Jansen, and S. Brinkkemper. Industry taxonomy engineering: the case of the european software ecosystem. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*, ECSA '10, pages 111–118, New York, NY, USA, 2010. ACM.

[5] John Klein, Ross Buglak, David Blockhow, Troy Wuttke, and Brenton Cooper. A reference architecture for big data systems in the national security domain. In *Proceedings of the 2nd International Workshop on BIG Data Software Engineering*. ACM/IEEE, 2016.

[6] Jiaheng Lu and Irena Holubová. Multi-model databases: A new journey to handle the variety of data. *ACM Computing Surveys*, 52(3), 2019.

[7] Loup Meurice and Anthony Cleve. Supporting schema evolution in schema-less nosql data stores. In *24th International Conference on Software Analysis, Evolution and Reengineering (SANER'17)*, pages 457–461. IEEE, 2017.

[8] Sergi Nadal and et al. A software reference architecture for semantic-aware big data systems. *Information and Software Technology*, 90:75–92, 2017.

[9] B.M. Pasquetto, I.V.and Randles and C.L. Borgman. On the reuse of scientific data. *Data Science Journal*, 16, 2017.

[10] Klaus Pohl, Günter Böckle, and Frank J. van der Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.