Herramienta para la evaluación de calidad de datos

Informe final de Proyecto de Grado

13 de febrero de 2019

Florencia Clerici
Beatriz Fernández

Tutor:
Adriana Marotta

Instituto de Computación
Facultad de Ingeniería - Universidad de la República
Montevideo - Uruguay

Resumen del trabajo

Durante los últimos años los datos han tomado una gran importancia en las organizaciones. En los procesos de producción, almacenamiento y utilización de los datos se generan problemas de calidad que repercuten de forma negativa en la operativa de las organizaciones. Por lo tanto, es de suma importancia conocer y mejorar la calidad de los datos mediante, la gestión de la misma.

En este proyecto se realiza un relevamiento de las herramientas existentes en el mercado y de los servicios disponibles en la web para realizar la medición de calidad de datos. En base a este análisis y la definición de requerimientos junto a la tutora se realizó el diseño, la implementación y documentación de una herramienta para la evaluación de la calidad de datos de bases de datos relacionales, la cual utiliza los conceptos de calidad de datos presentados en el curso de Calidad de Datos de FING [5]. Se logró un prototipo avanzado que puede ser utilizado por el resto de la comunidad académica.

La herramienta permite y asiste en la definición de un modelo de calidad de datos para una base de datos relacional. Luego permite la ejecución del modelo para medir la calidad y la visualización de los resultados.

También se deja disponible una biblioteca de especificaciones e implementaciones de métricas de calidad que pueden ser reutilizados. Las implementaciones de las métricas de calidad se realizan mediante web services externos a la herramienta, permitiendo la utilización de web services disponibles en la web.

Se realizó un caso de estudio sobre una base de datos real, pudiendo realizar la evaluación de los datos y detectar donde se encuentran los mayores problemas de calidad.

Índice general

1. Introducción

En las organizaciones los datos han tomado mucha importancia y diariamente se genera una gran cantidad de datos, de los cuales no se conoce la calidad. La calidad de datos es sumamente importante para poder utilizarlos de manera correcta y por lo tanto es necesaria la gestión de la calidad de datos.

La principal tarea de la gestión de calidad de datos es la evaluación de la calidad de datos, y existe la necesidad de contar con herramientas que manejen los principales conceptos. Estos conceptos son: modelo de calidad de datos, dimensión de calidad, factor de calidad, métrica, granularidad, agregación, entre otros. Las herramientas más conocidas no manejan estos conceptos.

Generalmente, cuando se realiza la medición de la calidad de datos, ésta se realiza de manera manual, realizando muchas tareas de forma repetitiva y desorganizada, por lo tanto, existe la necesidad de una herramienta que permita y asista en la definición de un modelo de calidad de datos para una base de datos relacional y permita la ejecución del modelo para medir la calidad y la visualización de los resultados.

1.1. Objetivos del proyecto

El objetivo del proyecto es desarrollar una herramienta de calidad de datos que cumpla lo siguiente:

- Ser flexible, para adaptarse a cualquier base de datos y para permitir definir cualquier modelo de calidad.
- Proveer una biblioteca de especificaciones e implementaciones de métricas de calidad que sea reutilizable.
- Permitir la invocación de web services externos a la herramienta que midan la calidad de datos.
- Facilitar el análisis de los resultados obtenido en la medición de la calidad de datos de una base de datos.

Para lograr este objetivo se comenzó por hacer un relevamiento de las herramientas existentes y los servicios disponibles en la web para realizar la medición de la calidad de los datos.

Como resultado del proyecto esperamos tener un prototipo de la herramienta de evaluación de calidad de datos con las características mencionadas anteriormente. Además, como parte de la documentación del proyecto se incluirá el manual de usuario de herramienta, el relevamiento realizado de las herramientas de calidad de datos existentes y un caso de estudio aplicado a una base de datos real.

1.2. Organización del documento

El informe consta de 6 capítulos, en el capítulo 2 “Conceptos básicos - Calidad de datos” se presentan los conceptos básicos de calidad de datos, de modelo de calidad y de la medición de la calidad de datos, necesarios para entender el marco teórico del proyecto.
En el capítulo 3 “Trabajos relacionados y herramientas existentes” se presentan los trabajos de calidad de datos existentes a nivel académico y las herramientas disponibles en el mercado. Además, evaluamos si cumplen o no con los objetivos del proyecto.
El análisis y el diseño de la herramienta se describen en el capítulo 4 “Análisis y diseño de la herramienta”, mientas que en el capítulo 5 “Implementación” se presentan las decisiones que se tomaron para solucionar los problemas que surgían durante la implementación y se describe la biblioteca (catálogo) implementada. Este capítulo también incluye la configuración de la herramienta y como se puede extender el proyecto a diferentes motores de base de datos.

Por último, en el capítulo 6 “Caso de estudio” se realiza el caso de estudio incluyendo el paso a paso de cómo utilizar la herramienta y los resultados obtenidos de la evaluación de una base de datos.

El informe cuenta con 6 anexos. En el anexo I “Casos de Uso” se especifican los casos de uso de las principales funcionalidades de la herramienta. En el anexo II “Extender a otro motor” se listan los métodos que son necesarios implementar para extender la herramienta a otros motores de base de datos.
En el anexo III “Definición del modelo de calidad del caso de estudio” se incluye el reporte obtenido de la exportación a pdf de la definición del modelo de calidad.
En el anexo IV “Reporte de ejecución de las métricas instanciadas del modelo de calidad datos del caso de estudio” se muestran los resultados obtenidos de la ejecución de las métricas instanciadas del caso de estudio. El anexo V “Reporte de ejecución de las agregaciones del modelo de calidad datos del caso de estudio” incluye los resultados de la ejecución de las agregaciones del caso de estudio.
En el anexo VI “Manual de instalación” se detalla paso a paso como instalar la herramienta.

2. Conceptos básicos - Calidad de datos

Definición de datos

Los datos representan objetos del mundo real en un formato que pueden ser almacenados, recuperados y elaborados por un procedimiento de software y comunicado a través de una red.

Definición de calidad de datos

El término calidad de datos es utilizado para referenciar un conjunto de características que los datos deben tener. [1] Por lo general cuando se piensa en calidad de datos, se reduce solo a la característica exactitud. Sin embargo, la calidad de datos es más que simple exactitud, es un concepto multifacético, donde existen diferentes dimensiones. [2]
En la literatura se utiliza el concepto “fitness for use” (aptitud para el uso), este enfatiza la importancia de tener en cuenta el punto de vista del consumidor sobre la calidad, en definitiva, es el consumidor quien va a juzgar si el producto está listo para el uso. Por lo tanto, define que los datos son de buena calidad cuando son aptos para el uso del consumidor. [4]
La medición de la calidad de los datos es comparar cuantitativamente entre una observación y el valor de referencia. Se realiza para brindar al usuario información sobre la calidad de los datos que se le va a entrega, mejorar la calidad de los datos y analizar su costo. [5]

Importancia de la calidad de datos

Durante los últimos años el mundo desarrollado se ha movido de una economía industrial a una economía de la información. La información es un material crítico para el éxito de las compañías, la calidad de datos pobre tiene serias consecuencias para la eficiencia y eficacia de las empresas. [3]
Data Warehousing Institute estima que los problemas de calidad de datos de los clientes les cuestan a las empresas de Estados Unidos más de $600 billones en un año en gastos de envío, impresión y gastos generales de personal, pero el costo real es mucho más alto. [3]
Las consecuencias de la escasa calidad de datos a menudo se experimentan en la vida cotidiana, pero por lo general, no se realiza una conexión con la causa. [2] Los problemas se pueden generar durante la producción de los datos, almacenamiento o utilización.

Dimensiones de calidad

Hay varios enfoques para encarar el estudio de la calidad de datos, en este trabajo vamos a utilizar el metamodelo de calidad presentado en [5].
Una **dimensión de calidad** de datos es un conjunto de atributos que representan un aspecto de calidad a alto nivel. Se puede ver como un agrupamiento de **factores de calidad** que tienen el mismo propósito. Un factor representa un aspecto particular de una dimensión de calidad. Cada uno puede medirse con diferentes **métricas**, las métricas son instrumentos que definen una forma de medir un factor de calidad. Para obtener un valor cuantitativo, se utilizan **métodos de medición**, los cuales implementan una métrica. De esta forma una métrica

puede ser medida por varios métodos.
La métrica se define a través de la semántica, la unidad de medición y la granularidad de la medida. La **granularidad de la medición** es la unidad básica de información a la que asociamos las medidas, en bases de datos relacionales las granularidades típicas son: celda, tupla, conjunto de celdas (vista), atributo/columna, tabla, grupo de tablas o base de datos (figura 2). En la figura 1 está representada la jerarquía de los conceptos de calidad.

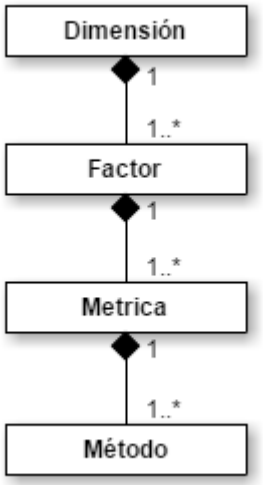


Figura 1: Diagrama de jerarquía de conceptos de calidad

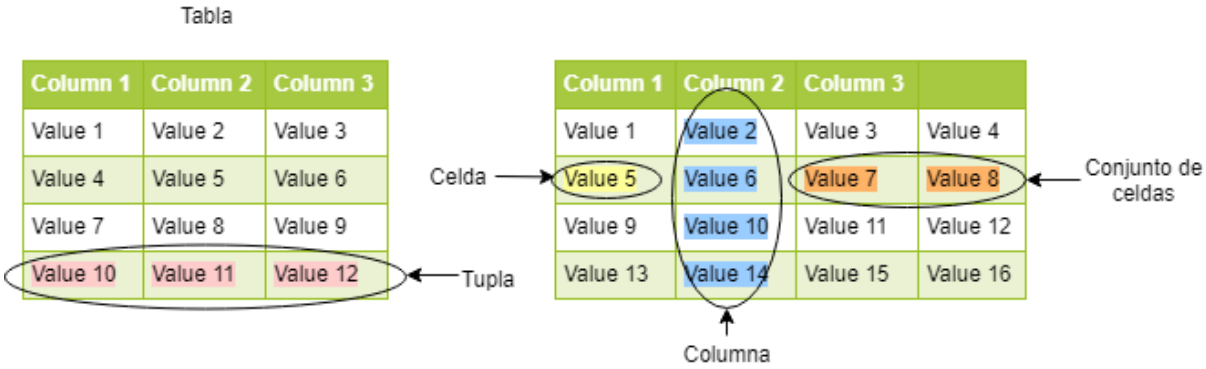


Figura 2: Granularidades

Otro concepto importante para la calidad de datos es el de **agregaciones**, el cual nos da un marco teórico para trabajar con los resultados de las métricas definidas. Dados los resultados de las mediciones se pueden realizar dos tipos de agregaciones. Uno de ellos es agregar sobre la granularidad de la medición, es decir, obtener a partir de una medida a un determinado nivel de granularidad, la medida a un nivel de granularidad menor. Por ejemplo, a partir de las medidas de las celdas, se pueden obtener las medidas para la tabla. La otra agregación posible es sobre los factores, dada más de una medición sobre los mismos datos con la misma granularidad, obtener una medida global de la calidad de los datos evaluados, pudiendo establecer un peso a cada factor, dependiendo de la importancia de este.

En las diferentes literaturas se definen variadas dimensiones [2, 3, 5, 7], en algunos casos con diferentes significados para un mismo nombre. No se ha establecido ningún conjunto como un estándar. [1] En este trabajo se considerarán las dimensiones sugeridas en [5]:

exactitud, completitud, frescura, consistencia y unicidad.

Exactitud (Accuracy)

Intuitivamente la exactitud indica qué tan precisos, válidos y libre de errores están los datos representados en el Sistema de Información (SI). Para exactitud se definen los factores correctitud semántica (semantic correctness), correctitud sintáctica (syntactic correctness) y precisión (precision).

La correctitud semántica indica si los datos en el SI corresponden a la realidad, es decir, mide que tan bien están representados los estados del mundo real. Verificar la correctitud semántica comparando los datos del SI con el mundo real es muy costoso. En general se compara con un referencial considerado como válido u otra base de datos.

La correctitud sintáctica indica si los datos del SI tienen errores sintácticos o de formato, interesa medir si los valores corresponden a valores válidos del dominio, sin importar si son valores reales o no. Para verificar la correctitud sintáctica se compara con un diccionario que representa el dominio o se puede chequear si satisfacen reglas de sintaxis.

La precisión analiza si los datos del SI brindan suficiente detalle, por ejemplo, para evaluar la precisión de una dirección tenemos que determinar si tienen suficiente detalle. La dirección “J. Herrera y Reissig 565, 11300, Montevideo” es más precisa que “Montevideo”, si lo que queremos representar es la dirección de la facultad en nuestro sistema de información.

Completitud (completeness)

Indica si el SI contiene toda la información de interés, recubre aspectos extensionales (cantidad de entidades/estados de la realidad representados) e intencionales (cantidad de datos sobre cada entidad/estado) del SI. Para la completitud se definen los siguientes factores: cobertura (coverage) y densidad (density).

La cobertura es la porción de los datos de la realidad contenidos en el SI, lo cual implica una comparación del SI con el mundo real que suele ser costosa. Como alternativa se plantea estimar el tamaño que tendría un referencial.

La densidad mide cuánta información se tiene sobre las entidades del SI.

Frescura (freshness)

La dimensión frescura involucra una perspectiva temporal de los datos del SI, es decir, que tan viejos son. Los factores que se definen para la frescura son: actualidad (currency), oportunidad (timeliness) y volatilidad (volatility).

El factor actualidad mide el desfase entre los datos del SI y los datos reales, es decir, que tan vigente son los datos del SI.

Para el factor oportunidad se tiene en cuenta si los datos están actualizados en el momento que se necesita.

Por último, el factor volatilidad mide el lapso que los datos se mantienen válidos.

Consistencia (consistency)

La consistencia captura la satisfacción de reglas semánticas definidas sobre los datos, dichas reglas pueden ser reglas de integridad para una base de datos o reglas de usuarios. Los

factores que se definen para la dimensión consistencia son: integridad de dominio (domain integrity), integridad intra-relación (relation integrity) e integridad interrelación (inter-relation integrity), en todos los casos mide que tan bien se satisfacen las reglas de integridad.

Unicidad (uniqueness)

La dimensión unicidad indica el nivel de duplicación entre los datos del SI, lo cual ocurre cuando una misma entidad está representada dos o más veces. Los factores que se definen son no-duplicación (duplication-free) y no-contradicción (contradiction-free), en ambos casos interesa medir la cantidad de repetidos.

Modelo de calidad

Para poder tratar la calidad de un SI de manera ordenada es necesario definir un modelo de calidad adecuado a las necesidades y prioridades de los consumidores de dicho SI. Definir un modelo de calidad implica determinar y especificar las dimensiones y factores a medir, las métricas y datos a los cuales se van a aplicar y la base de metadatos que se utilizará para almacenar los resultados de los procesos de dicha medición.
Las **métricas instanciadas** contendrán una referencia a los atributos de la base de datos sobre el cual se le aplicará la métrica, una referencia a la métrica a aplicar y al método de medición.

Es importante determinar las dimensiones y los factores de calidad que vamos a medir, ya que puede ser muy costoso medir todas las dimensiones y factores para todos los datos, por eso es necesario priorizarlos. Para realizar dicha priorización se pueden utilizar algunos de los siguientes métodos: relevamiento de los requerimientos de usuarios, perfil de los datos (Data Profiling) [9] o Goal-Question-Metric [8].

El relevamiento de los requerimientos de usuarios se aplica de dos formas, a partir de los datos prioritarios se definen las dimensiones, los factores y las métricas o primero se definen las dimensiones y factores, se seleccionan los datos y por último se definen las métricas.

El perfil de los datos brinda noción de donde están las fallas de calidad, a partir de esta se definen las dimensiones, factores y métricas, sin la intervención de los usuarios finales. Consiste en evaluar los datos disponibles y obtener estadísticas e información sobre los mismos. Existen diversos métodos para evaluar los datos y producir metadatos, obtener estadísticas, determinar metadatos que involucran varias columnas y técnicas para detectar propiedades aproximadas o condicionales del conjunto de datos. El perfil de datos permite obtener conocimientos de la estructura, las relaciones, el volumen, los problemas y su frecuencia y los patrones que cumplen los datos del SI a evaluar. Ejemplos de estas técnicas son las estadísticas básicas, el análisis de metadatos, el análisis de patrones o la detección automática de foreign key.

El método Goal-Question-Metric (GQM) es un paradigma de diseño de SI que utiliza el enfoque top-down para definir los factores que interesan medir, identificando primero los problemas de calidad para luego determinar que métricas son las apropiadas para

cuantificarlos. GQM define tres niveles de abstracción: conceptual, operacional y cuantitativo. El nivel conceptual consta de identificar un conjunto de objetivos de calidad de alto nivel que apuntan a resolver problemas de calidad de la organización. En el nivel operacional se toma cada uno de los objetivos definidos en el nivel anterior y se derivan preguntas que los definan tan completos como sea posible, caracterizando la manera de alcanzar los objetivos. En el nivel cuantitativo se definen las métricas necesarias para responder las preguntas de forma cuantitativa y para realizar un seguimiento de la conformidad del producto o proceso respecto a los objetivos. Por último, se determina el procedimiento de recolección de los datos, incluyendo los mecanismos de validación y análisis.

Para la definición de las dimensiones y factores, no nos podemos olvidar que los factores no son independientes entre sí, por ejemplo, los errores de tipeo (correctitud sintáctica) en un dato puede provocar que sea considerado semánticamente incorrecto por no encontrarlo en un referencial. La dependencia entre los factores permite que se pueda estimar un factor a partir de otro, al mejorar un factor se puede afectar otro factor de calidad de forma positiva o negativa dependiendo del contexto de aplicación. En la correlación positiva se mejoran ambos factores y en la negativa se mejora un factor mientras que empeora otro.

Medición de la calidad de los datos

La forma de medir la calidad de datos depende de la dimensión/factor de calidad, la métrica elegida, el tipo de la fuente de datos a los que se le aplica la medición, si la medición se realiza online u offline.
La medición online se realiza al momento de extraer los datos para dárselo al usuario, mientras que offline es en un momento distinto al que se usan los datos.

Previo a la medición se define el factor de calidad, el objeto sobre el cual se va a medir, la granularidad, la métrica, el proceso de medición (implementación y ejecución de la medida), las agregaciones para cambiar de granularidad y el modelo de datos para almacenar los valores de la calidad obtenidos (metadata).

En la literatura, los métodos para evaluar la calidad de datos se clasifican en objetivos y subjetivos.
El enfoque objetivo se basa en que el conjunto de los datos es dependiente o independiente de las tareas. El enfoque independiente refleja los estados de los datos sin un conocimiento de la aplicación, se aplica a cualquier conjunto de datos independientes de las tareas. En el dependiente se desarrollan en contextos de aplicaciones concretas incluyendo las reglas de negocio de la organización y las regulaciones gubernamentales. [6]
El enfoque subjetivo refleja las necesidades y experiencias de los individuos involucrados con los datos, por ejemplo, los consumidores de los datos. Los datos son comparados con datos de la realidad.
Pipino [6] propone una metodología para aplicar los enfoques objetivo y subjetivo para determinar la calidad de los datos. El primer paso consta en realizar evaluaciones objetivas y subjetivas de la calidad de los datos. Luego se comparan los resultados, identificando las discrepancias y determinando sus causas. Por último, se determinan y toman las acciones necesarias para mejorar la calidad.

AIMQ, una metodología para la evaluación de la calidad de la información utiliza el enfoque

subjetivo. Está compuesta de un modelo de calidad de información que describe lo que significa la calidad para los consumidores y administradores, un cuestionario para medir la calidad de datos de las dimensiones determinadas en el paso anterior y técnicas para interpretar las mediciones. [11]
[pierce04] [aimq02]

Llamamos **metadatos de calidad** a la representación de las dimensiones de calidad y las medidas de calidad que aplicaremos sobre un modelo. Con dicho objetivo se proponen extensiones a los modelos tradicionales para base de datos, para poder representar adecuadamente los aspectos relacionados con las dimensiones de calidad. Hay que representar los conceptos generales de calidad (dimensiones, factores y métricas), la instanciación de las métricas y la granularidad aplicada. En este caso se debe guardar el valor obtenido en la medición y la referencia al dato.

Como vimos al principio el objetivo es tener una mejor calidad de datos en los SI, para reducir costos asociados a errores de datos, lo que vimos hasta ahora son algunas de las tareas que hay que realizar para cumplir dicho objetivo. En la figura 3 se muestra el proceso completo con todas las tareas. [5]

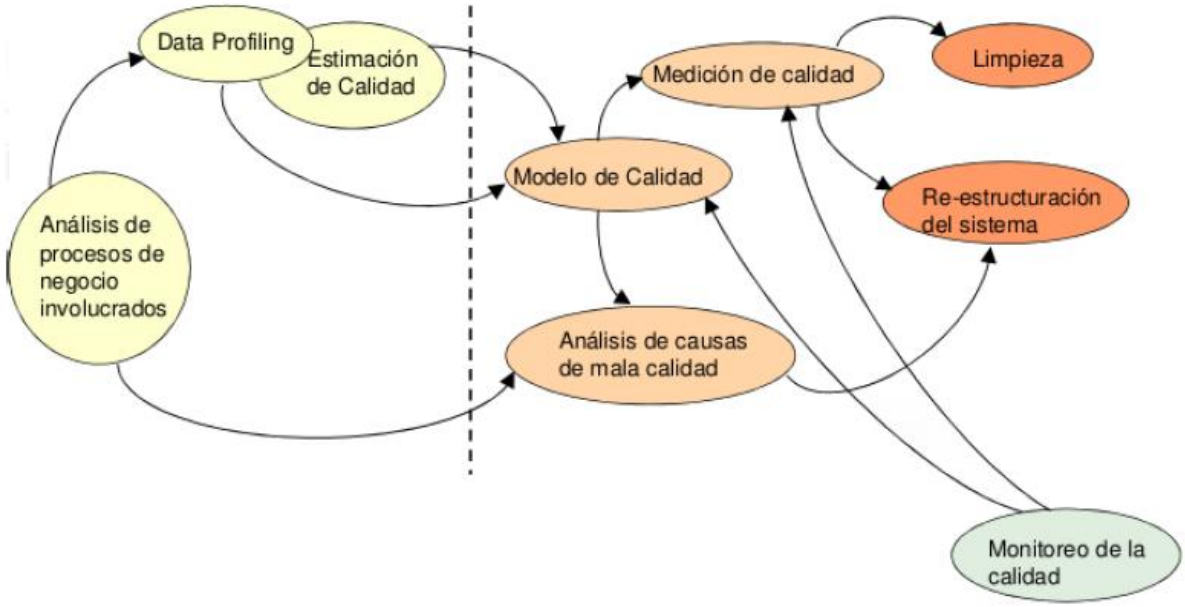


Figura 3: Tareas de la gestión de la calidad en SI [5]

En primer lugar, hay que analizar los procesos de negocio involucrados para conocer el dominio del SI. Como vimos un posible camino a seguir para definir el modelo de calidad de datos es hacer un perfil de los datos, para obtener rápidamente una estimación de la calidad y tomarlo como dato de entrada para definir el modelo de calidad. El análisis de causas de mala calidad es muy importante ya que se pueden detectar errores sistemáticos que se están dando al usar el sistema o procesar los datos, para esto se deben tener en cuenta los procesos. Las tareas de definir el modelo de calidad y realizar las mediciones las vimos previamente. La reestructuración del sistema es sumamente importante para solucionar los problemas sistemáticos detectados tanto en el análisis de causas de mala calidad como en la medición de calidad. La limpieza de datos se realiza una vez que se hizo la medición de calidad del SI, con el fin de mejorar la calidad. La tarea de monitorear la calidad no es menor,

se basa en ir tomando medidas de calidad a lo largo del tiempo y poder realizar comparaciones de la calidad antes y después de tomar determinadas acciones (reestructuración del sistema y/o limpieza). Además, se puede ir redefiniendo el modelo de calidad si se cree necesario.

3. Trabajos relacionados y herramientas existentes

En este capítulo del informe revisamos los trabajos relacionados con la calidad de datos a nivel académico e investigamos las herramientas que ofrece la industria para la gestión de la calidad de los datos. Ponemos énfasis en el manejo del modelo de calidad de datos.

3.1. Evaluación de la calidad de datos - trabajos académicos

QBox-Service [12]

En el mercado existen diversas herramientas de calidad, las cuales proveen funcionalidades individuales sobre la evaluación de la calidad, contienen diversos modelos y patrones de acceso sin proveer mecanismos de interoperación, manejan diferentes conceptos de calidad, en diferentes niveles de abstracción y expresados en terminología ad-hoc. La plataforma QBox-Services permite la interoperabilidad entre estas herramientas de calidad. Con esta herramienta es posible realizar un análisis multidimensional ya que permite realizar agregaciones entre medidas obtenidas en diferentes observaciones.

Esta plataforma utiliza el enfoque GQM, brinda un catálogo unificado de conceptos de calidad, permitiendo de esta forma la reutilización de los métodos de calidad y la interoperabilidad de diversas herramientas. QBox-Service está compuesto por varios módulos: QBox-Foundation, QMediator y QManagement.

El módulo QBox-Foundation provee las funcionalidades para especificar el modelo basado en el paradigma GQM. Este permite definir objetivos en alto nivel y elegir un conjunto de factores de calidad y métricas que caracterizan la forma de evaluar cada objetivo. Las métricas son elegidas, refinadas y especializadas desde un metamodelo de calidad general, el cual constituye una librería de conceptos de calidad. Además, la personalización y unión de funcionalidades ayudan al experto a derivar desde un metamodelo de calidad genérico a un Modelo Personalizado de Calidad (PQM).

QMediator es el módulo que provee las funcionalidades para encontrar el servicio adecuado a los requerimientos, habilitando su ejecución y retornando los resultados. Este actúa como mediador entre los requerimientos expresados en PQM y los servicios disponibles.

Por último, el QManagement ejecuta los servicios de calidad para un objetivo específico y provee una interfaz para analizar los resultados. Los resultados son almacenados en una base de datos en forma de estrella que favorece la agregación de las mediciones, la computación de indicadores complejos y el análisis de las correlaciones entre las medidas.

Esta herramienta cumple con el objetivo de este proyecto de reutilizar la implementación de métricas de calidad y el análisis de las medidas de calidad obtenidas. Lo que falta contemplar en esta herramienta es la definición del modelo de calidad como se describió en el Capítulo 2 y la generación de una base de metadatos de calidad para almacenar las medidas de

calidad obtenidas. Además, tuvimos la oportunidad de reunirnos en facultad con Laura González, una de las responsables del proyecto que nos comentó que se realizó la implementación de un prototipo de la solución, pero actualmente no se encuentra disponible.

Detecting Data Errors: Where are we and what needs to be done?

"Detecting Data Errors: Where are we and what needs to be done?" [18] es un artículo del 2016 que nos pareció importante tener en cuenta en el informe ya que nos dio ideas para el proyecto, por lo tanto, a continuación, lo analizamos en profundidad.

Existen diversas herramientas para detectar y posiblemente reparar ciertas clases de errores como *outliers, duplicates, missing values y violations of integrity constraints*. En un mismo conjunto de datos pueden coexistir diferentes tipos de errores, esto hace que a menudo sea necesario ejecutar más de una herramienta. En este artículo los autores plantean dos preguntas: ¿Hay alguna herramienta lo suficientemente robusta para detectar la mayoría de los errores en los conjuntos de datos del mundo real? y ¿Cuál es la mejor estrategia para optimizar la detección de errores utilizando múltiples herramientas de manera integral?

Para responder estas preguntas decidieron seleccionar ocho herramientas de limpieza de datos. Estas herramientas utilizan varias técnicas de detección de errores y son capaces de detectar algún tipo de error de los que se analizan en el artículo. Además, eligieron cinco conjuntos de datos reales.

Actualmente las soluciones y las herramientas se pueden clasificar en cuatro categorías:

- Algoritmos de detección basados en reglas que definen conjuntos de reglas que los datos deben cumplir.
- Aplicación de patrones y herramientas de transformación los cuales descubren los patrones que cumplen los datos (sintácticos y semánticos) y los utiliza para detectar errores.
- Algoritmos de detección de error cuantitativa que exponen valores atípicos y "glitches".
- Agrupación de datos y algoritmos de duplicación que realizan la consolidación de entidades cuando varios registros tienen datos para la misma entidad.

Al considerar las herramientas surgieron los siguientes desafíos:

1. Síntesis de datos y errores: la evaluación de las herramientas y algoritmos fue realizada en un "ambiente controlado". Por lo tanto, no está claro cómo se comportará con datos reales y no de prueba.
2. Combinación de tipo de errores y de herramientas: al considerar un solo tipo de algoritmo en el estudio, se pierde la oportunidad de acumular evidencia de múltiples herramientas.
3. Minimizar intervención humana: la mayoría de las herramientas requieren la intervención del humano, por ejemplo, para verificar los errores detectados o para especificar reglas de limpieza. Por lo tanto, es necesario minimizar la intervención humana. Al combinar herramientas y ordenar su ejecución es posible minimizar la intervención humana lo cual requiere evaluar la interacción entre las herramientas.

Los autores configuraron las herramientas de forma iterativa eligiendo los mejores parámetros o definiendo reglas de calidad razonables. En todos los casos utilizaron las características que proveen y no escribieron código.

Por un lado, evalúan las ejecuciones individuales de cada herramienta y luego analizan diferentes métodos para combinarlas. Los métodos que utilizaron fueron: *Union all*, *Min-k*, *Ordering based on precision*. *Union all* toma la unión de los errores emitidos por todas las herramientas. *Min-k* considera solo los errores detectados por al menos k herramientas y excluye aquellos detectados por menos de k herramientas.
Ordering based on precision determina el orden en el cual deben ser evaluadas las herramientas. Los desempeños de las herramientas pueden ser medidos en base a la precisión y la sensibilidad (recall). La precisión se define como la fracción de los errores marcados correctamente y la sensibilidad como la fracción de errores reales descubiertos por la herramienta (figura 4). Para determinar la sensibilidad es necesario conocer todos los errores presentes en los datos, lo cual es imposible en un nuevo conjunto de datos, mientras que la precisión es más fácil de estimar. Consideran que el costo de que un humano compruebe un error detectado sobre el valor de identificar un error real tiene que ser mayor a la precisión, en caso contrario el costo de comprobar será mayor al valor obtenido y la herramienta no será ejecutada. Para determinar el orden de ejecución también se considera que los errores detectados por las diferentes herramientas pueden ser detectados por otras de mayor precisión.

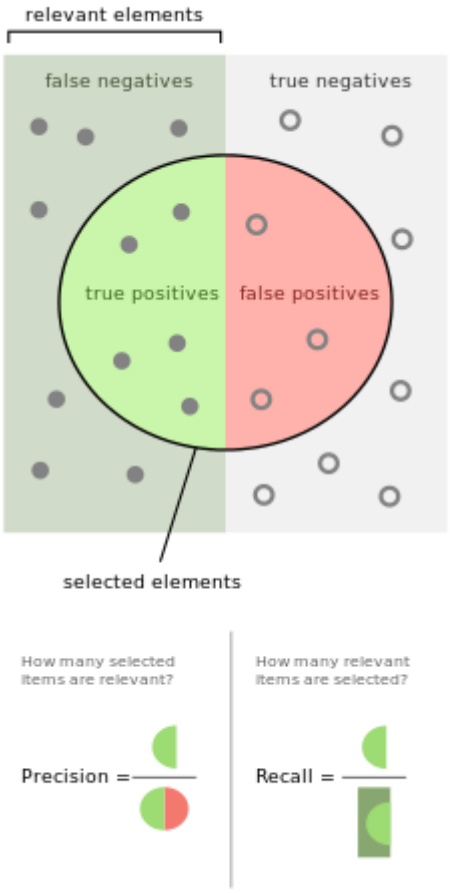


Figura 4: Definición de precisión y sensibilidad [38].

Luego de ejecutar los métodos obtienen ciertas observaciones. Union all aumenta la sensibilidad, pero disminuye la precisión debido a los falsos positivos. El método *Min-k* al aumentar la cantidad de herramientas (k) mejora la precisión, pero reduce la sensibilidad. En el caso de *Ordering based on precision* si la prioridad es la sensibilidad, lo mejor es empezar con un umbral conservador, siendo el umbral el radio entre el costo total y el valor total obtenido, y para minimizar el trabajo humano se puede sacrificar la sensibilidad.
Como conclusiones del experimento se tiene que no hay una herramienta dominante que pueda detectar todos los errores para cualquier conjunto de datos y seleccionar el orden correcto en el que se aplican las herramientas puede mejorar la precisión reduciendo el trabajo humano de verificación. Los resultados de las herramientas basadas en reglas y las que detectan duplicados pueden mejorar si se enriquecen los datos, como por ejemplo resolver casos de datos ambiguos.
Los lineamientos que plantean para continuar trabajando son seguir investigando cual es la mejor metodología para combinar las distintas herramientas y tratar de manera integral la detección de errores. El enriquecimiento de datos se considera un factor importante para mejorar las detecciones de las herramientas. Dada la importancia que tiene el usuario en el proceso de detección de errores, es fundamental brindarle herramientas para que comprenda mejor lo que se está analizando y pueda tomar mejores decisiones. Por último, trabajar con errores sintácticos no ayuda a determinar la utilidad de una herramienta en el mundo real, por lo tanto, es deseable realizar este estudio y obtener conclusiones para conjuntos de datos “reales”.

3.2. Herramientas comerciales para la evaluación de la calidad de datos

Se realizó un relevamiento de las herramientas existentes en el mercado, y se eligieron algunas para realizar una evaluación, así como también se realizó la evaluación de una herramienta académica. El resultado de dicha evaluación se presenta en la tabla 1.

Herramienta	**Dimensión**	**Factor**	**Granularidad**	**Dominio**
DataCleaner	Exactitud	Correctitud sintáctica	Celda	
		Correctitud semántica	Conjunto de celdas	Direcciones
		Precisión	Celda	Depende de los datos
	Consistencia	Integridad de dominio	Conjunto de celdas	
		Intra-relación	Celda	
		Inter-relación	Celda	
	Completitud	Densidad	Conjunto de celdas	
	Unicidad	No-duplicación	Conjunto de celdas	

	Frescura	Actualidad	Celda	Depende de los datos
Data Quality Services	Exactitud	Correctitud sintáctica	Celda	
	Unicidad	No-duplicación	Conjunto de celdas	
Open Studio for Data Quality de Talend	Exactitud	Precisión	Atributo - Celda	Depende de los datos
		Correctitud sintáctica	Atributo - Celda	
	Completitud	Cobertura	Conjunto de atributos - Conjunto de celdas	Depende del referencial de datos
		Densidad	Conjunto de atributos - Conjunto de celdas	
	Consistencia	Integridad de dominio	Atributo - Celda	
		Intra-relación	Conjunto de atributos - Conjunto de celdas	
		Inter-relación	Conjunto de atributos - Conjunto de celdas	
	Unicidad	No-duplicación	Conjunto de atributos - Conjunto de celdas	
		No-contradicción	Conjunto de atributos - Conjunto de celdas	
	Frescura	Actualidad	Conjunto de atributos - Conjunto de celdas	
Oracle Enterprise Data Quality	Exactitud	Precisión	Celda	Depende de los datos
		Correctitud sintáctica	Atributo - Celda	
		Correctitud semántica	Conjunto de celdas	Direcciones
	Completitud	Densidad	Conjunto de atributos - conjunto de celdas	
	Consistencia	Integridad de dominio	Conjunto de atributos - conjunto de celda	
		Intra-relación	Conjunto de atributos - conjunto de celdas	
	Unicidad	No-duplicación	Conjunto de atributos - conjunto de celdas	
		No-contradicción	Conjunto de atributos - conjunto de celdas	

	Frescura	Actualidad	Conjunto de celdas	Depende de los datos

Tabla 1: Evaluación de las herramientas de calidad de datos

DataCleaner es una herramienta de escritorio desarrollada en Java, las pruebas las realizamos en la edición profesional 5.0.3, que es la versión que ofrecen de forma gratuita por un mes. Además, DataCleaner es open source, el código está disponible en GitHub y lo lleva adelante Neopost and Human Inference. Al ser open source tiene la flexibilidad para agregar nuevos componentes. Si bien hay documentación en la aplicación y un manual en la web, no describe todos los parámetros de configuración y no tiene ejemplos. Además, la comunidad no es muy grande y no es una herramienta de uso masivo, lo que hace que ante algunos errores no se encuentre información al respecto.
Permite conectarse a diversas bases de datos: archivo .csv, archivo .xls y xlsx, archivo .mdb, SAS library, dBase, archivo xml, archivo json, salesforce, sugarCRM, DataHub, MongoDB, CouchDB, Elastic Search Index, Cassandra, Hbase, Neo4j, Apache Hive, PostgreSQL, Oracle, Microsoft SQL Server, Apache Derby, H2, Hsqldb/HyperSQL, Sybase, Cubrid, DB2, Firebird, Ingres, LucidDB, MySQL, Pervasive, SAP DB, SQLite y Teradata.
Tiene un conjunto de analizadores que mediante la combinación de estos nos permiten definir algo parecido a un modelo de calidad, se pueden definir una o más métricas y los resultados de las mediciones se pueden guardar en Excel.
No permite definir métricas directo en SQL, si permite hacerlo en JavaScript. Algunos analizadores permiten la interacción con servicios web “especiales” para verificar direcciones, mails y teléfonos. También permite identificar duplicados y hacer limpieza de datos. De una forma muy sencilla, casi automática, se puede realizar un perfil de los datos.
No maneja los conceptos que definimos de dimensiones y factores, si bien genera medidas de calidad, no se puede definir la granularidad de la medición, está implícita en el analizador que se utilice. Los resultados no se pueden agregar (pasar de una granularidad de resultado celda a una granularidad columna) y a partir de los resultados de todos los analizadores utilizados no podemos obtener un valor de la calidad global. [16, 17]

Data Quality Services es una herramienta de escritorio disponible en Microsoft SQL Server desde su edición 2012. Las pruebas se realizaron en la versión enterprise 2016. Permite detectar duplicados, limpiar, validar, y unir, usando un referencial únicamente utilizando SQL Server. [14, 15]

La herramienta Open Studio for Data Quality de Talend es una aplicación de escritorio open source con licencia apache, permite realizar data profiling y evaluar la calidad de datos, nos centraremos en la evaluación de la calidad de datos. La herramienta no utiliza los conceptos de dimensiones, factores ni métricas de calidad, así como tampoco la misma terminología utilizada para este proyecto respecto a la calidad de datos.
Los análisis retornan las medidas con diferentes granularidades, dependiendo del método utilizado la granularidad es atributo, conjunto de atributos o tabla, en algunos casos es posible visualizar los registros que cumplen o no las condiciones de la métrica. En la mayoría de los casos además de mostrar la cantidad brindan el porcentaje de registros que cumplen la condición respecto al total analizados. No es posible definir agregaciones entre diferentes métricas ni obtener un análisis global de la base de datos.
Tiene métricas preestablecidas y permite definir métricas llamadas User Indicator Definition

(indicadores) especificadas en lenguaje SQL o en Java. Los indicadores en Java se definen en un archivo jar externo a la herramienta, mientras que las consultas SQL se definen en la interfaz de Talend. El nombre de la tabla y la columna que se va a utilizar en el indicador se seleccionan al momento de utilizarlas en un analizador. Los indicadores en lenguaje SQL permiten utilizar diferentes columnas de una tabla en las consultas. Estos permiten determinar medidas de calidad para diferentes dimensiones y factores, se realizaron diferentes pruebas teniendo como resultado medidas de consistencia (integridad intra-relación e integridad de dominio), exactitud (correctitud sintáctica, verificado por extensión y por compresión) y unicidad (no duplicación y no contradicción). La granularidad es columna y permite ver las filas que cumplen la cláusula where de la consulta.
Las fuentes de datos son bases de datos o archivos de texto delimitados. Soporta conexión a las bases de datos Microsoft SQL Server, Oracle, PostgreSQL, MySQL, AS400, Exasol, General JDBC, Hive, IBM DB2, IBM DB2 ZOS, Impala, Informix, Ingres, Notezza, Oracle Custom, Oracle OCI, Oracle with SID, Oracle with service name, Redshift, SqlLitle, Sybase, Teradata y Vertica.
Brinda buenos tutoriales paso a paso para comenzar a utilizar la herramienta, así como la descripción de cada analizador.
Talend ofrece bajo suscripción Talend Data Management Platform, además de las características de Talend Open Studio for Data Quality permite definir procesos batch, enmascaramiento de datos, herramientas de administración y consola de monitoreo. [13] Para realizar la evaluación se utilizó la versión 6.2.0 de Talend Open Studio for Data Quality.

La familia de productos Oracle Enterprise Data Quality ofrece profiling, auditoría y permite obtener ciertas medidas de calidad de datos [Tabla 1]. La licencia es gratuita y analizamos el componente Director versión 12.2.1.0.0.
Los análisis están predefinidos, el usuario selecciona sobre que tabla y atributos se aplicará y genera una medida de calidad. La granularidad es predefinida para cada análisis, en algunos casos retorna el resultado en varios niveles de granularidad y se muestran los registros que cumplen o no las condiciones. Solo permite utilizar atributos de una misma tabla en cada analizador y no brinda un análisis global de la base de datos. Es posible definir datos de referencia para utilizar en los analizadores. La salida de un analizador se puede exportar a Excel.
Es posible crear procesos que son secuencias de analizadores, donde la salida de un analizador (por ejemplo: datos que superan la evaluación) es la entrada del siguiente.
Permite definir script para especificar procesos lógicos en JavaScript o Groovy. El script solo permite retornar un valor por cada registro evaluado, no permitiendo realizar agregaciones. No brinda funcionalidades para definir consultas SQL.
Los procesos se exponen como web services, no fue posible consumirlo desde herramientas externas a Oracle Data Quality.
Los datos se obtienen de bases de datos, archivos de texto o xml, archivos MS Office: Access y Excel o conexiones JDBC. Las bases de datos soportadas son: Oracle, PostgreSQL, MySQL, Microsoft SQL Server, Microsoft Access, Sybase, DB2, Origenes de datos JNDI y Apache Hive.
La flexibilidad de la herramienta está limitada a los métodos que se puedan escribir en los scripts. No utiliza la nomenclatura utilizada en este proyecto. Contiene tutoriales de cada producto y para cada analizador.

No encontramos herramientas que se encarguen de todas las etapas de la gestión de calidad de datos, las herramientas que investigamos resuelven una o más etapas, pero no realizan una gestión “integral” de la calidad de datos. En general son herramientas que permiten hacer un perfil de los datos, realizar limpieza de datos y/o medir algunos factores, pero se dificulta realizar un seguimiento de la calidad.
Si como modelo de calidad entendemos que se deben definir y especificar las dimensiones y los factores a medir, las métricas y los datos a los cuales se aplicarán, la base de metadatos de calidad donde se registran los resultados de la medición, entonces las herramientas que vimos no permiten definir el modelo de calidad completo, ya que en ninguna se puede definir ni especificar las dimensiones y los factores que se van a medir, y solo algunas permiten guardar o exportar los resultados en algún formato.
Al no poder definir la dimensión ni el factor que vamos a medir, lo que estamos haciendo es definir las métricas que vamos a ejecutar con una granularidad asociada que no se puede modificar. A nivel de los resultados la mayoría de las herramientas (todas menos Oracle) no permiten realizar agregaciones sobre los resultados de una misma métrica (por ejemplo, pasar de un resultado de celda a uno de columna). Con ninguna de las herramientas es posible obtener un resultado de calidad de todo el modelo, es decir, podemos especificar y medir varios aspectos de la calidad de los datos, pero no podemos definir una forma de agregar esos resultados para llegar a un valor de calidad para los datos analizados.

En la tabla 2 se muestran los resultados obtenidos de los principales criterios evaluados.

	DataCleaner	Data Quality Services	Open Studio for Data Quality de Talend	Oracle Enterprise Data Quality
Permite definir modelo de calidad	No	No	No	No
Agregaciones	No	No	No	Si
Permite consumir web services	Si	No	No	No
Métricas en diferentes granularidades	No	No	No	En algunos casos retorna el resultado en varias granularidades
Flexibilidad para definir nuevas métricas	Si, en JavaScript	Si	Si, SQL y Java	Si, JavaScript o Groovy

Tabla 2: Resumen de la evaluación realizada a las herramientas de calidad de datos

4. Análisis y diseño de la herramienta

En base al estudio teórico de los conceptos de calidad de datos, entendemos que la forma correcta de encarar un problema de este tipo implica mucho más que el análisis de los datos y corrección de los mismos. Es necesario especificar un proceso que permita definir, medir, corregir y monitorear la calidad de un conjunto de datos. En este capítulo vamos a realizar el análisis y diseño de la herramienta que vamos a implementar.

Las herramientas existentes tienen resueltos algunos puntos intermedios del proceso, pero ninguna incorpora el concepto de modelo de calidad como lo planteamos en el capítulo 2. Creemos que poder definir y manejar un modelo de calidad es indispensable para la gestión sistemática de la calidad de los datos. Entendemos que hace falta incorporar a la herramienta el marco teórico para poder definir un modelo de calidad que determine qué aspectos nos interesa medir de los datos. Con lo anterior bien definido, luego del análisis y corrección es posible realizar un monitoreo continuo de la evolución de los datos. Este monitoreo es importante ya que analizar y corregir grandes cantidades de datos es costoso. Por lo tanto, es imprescindible poder cuantificar si las acciones que estamos tomando tienen el desempeño esperado.

En el marco del proyecto, encontramos como oportunidad desarrollar KiuPlus, una herramienta que permita definir tanto los criterios de calidad teóricos a utilizar como el modelo de calidad para un conjunto de datos, ejecutar los modelos definidos y analizar los resultados obtenidos, tratando de aprovechar los avances que hay actualmente en la industria.

En este capítulo se presentará el proceso realizado para la construcción de la herramienta. Primero se hace el análisis del problema planteado. Posteriormente se describen las decisiones de diseño que tuvimos que tomar, en base a los problemas y restricciones que fueron surgiendo.

1.1. Análisis

En esta sección abordaremos el análisis realizado previo a la implementación. Hacemos un breve repaso del problema que pretendemos resolver con la herramienta, identificamos los conceptos que nos interesa modelar, describimos las principales funcionalidades y detectamos cuáles son las restricciones que tenemos.

i. Resumen del problema

En primer lugar, identificamos las principales características que tiene que tener la herramienta a desarrollar para cumplir con el objetivo que nos planteamos en el proyecto. La misma debe brindar las funcionalidades necesarias para definir las dimensiones, los factores, las métricas, los métodos y los datos a los cuales se les va a medir la calidad, es decir el modelo de calidad.
Se conformará un catálogo con las definiciones de dimensiones, factores, métricas, métodos y la implementación de algunos métodos genéricos, de esta forma se podrán utilizar en varios modelos de calidad. De esta manera, la herramienta será más útil y flexible.

La definición del modelo de calidad se realizará en la interfaz de la aplicación y los métodos se implementarán a través de web services que serán invocados por la aplicación.
Para cada ejecución del modelo de calidad se generará automáticamente la base de metadatos de calidad donde se almacenarán las medidas obtenidas.
Finalmente permitirá la evaluación de los resultados, para ello se mostrarán los valores permitiendo realizar agregaciones.

ii. Enumeración de requerimientos

En esta sección enumeramos los principales requerimientos que debe cumplir la herramienta a desarrollar:

- Definir las dimensiones, los factores, las métricas y los métodos que se utilizarán para medir la calidad de una base de datos.
- Definir el modelo de calidad para una base relacional.
- Ejecutar el modelo de calidad.
- Generar la base de metadatos de calidad.
- Guardar en la base de metadatos de calidad los resultados de una ejecución.
- Realizar agregaciones de los resultados de las ejecuciones.
- Visualizar, exportar e imprimir el modelo de calidad de una base de datos.
- Visualizar, exportar e imprimir los resultados de una ejecución.

iii. Algunos conceptos por resolver - Enfoque

Antes de entrar en detalle en los requerimientos de la herramienta, nos parece importante explicar cómo vamos a adaptar algunos conceptos teóricos al uso de la herramienta.

Para definir el modelo de calidad, utilizando los conceptos teóricos vistos anteriormente, en la herramienta se van a instanciar métricas. El proceso de instanciar una métrica conlleva dos definiciones importantes. Una es definir qué métrica y qué método vamos a utilizar al momento de la evaluación. Por la forma que modelamos los conceptos, al realizar dicha definición queda determinado cuál será el factor y dimensión de la métrica instanciada. La otra definición importante es determinar a qué datos le vamos a aplicar la métrica (elemento a evaluar). Por lo tanto, el modelo de calidad de una base de datos va a ser el conjunto de métricas instanciadas que sean definidas. En la figura 5 diagramamos lo expuesto.

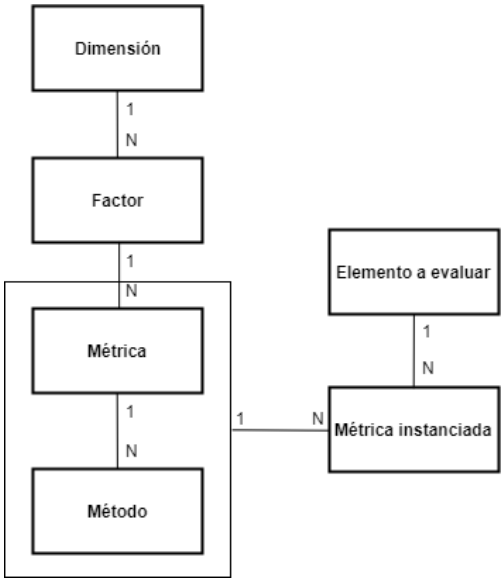


Figura 5: Métrica instanciada

Para facilitar el análisis de las medidas de calidad obtenidas se podrán realizar agregaciones. Definimos dos subconjuntos de granularidades, el subconjunto 1 está compuesto por las granularidades: celda, conjunto de celdas y tupla y el subconjunto 2 por conjunto de columnas, tabla y base de datos. Las agregaciones permiten cambiar la granularidad del resultado de una métrica instanciada del subconjunto 1 al 2.

A partir del artículo “Detecting Data Errors: Where are we and what needs to be done?” [18] comentado anteriormente decidimos considerar un nuevo concepto y es el de **métrica instanciada compuesta**. Es la combinación de un conjunto de métricas instanciadas, permitiendo obtener un resultado único para la ejecución de varias métricas instanciadas. Las métricas instanciadas que forman la métrica instanciada compuesta deben estar aplicadas al mismo conjunto de datos y deben tener granularidades compatibles. Llamamos granularidades compatibles a las granularidades de los mismos subconjuntos que definimos anteriormente. El objetivo de las métricas instanciadas compuestas es reutilizar los resultados de las ejecuciones de las métricas instanciadas y las operaciones que hayamos definido anteriormente.
Por ejemplo, si nos interesa medir la calidad de los teléfonos de una base de clientes de Montevideo, específicamente que cumplan algunas restricciones de dominio. En nuestro catálogo tenemos operaciones básicas como el largo, es decir, verifica que el string sea de un lago específico, y tenemos otra operación que verifica el prefijo de un string. Podemos combinar esas operaciones, sin necesidad de implementar una nueva, y verificamos que los teléfonos sean de largo ocho y que comiencen con dos.
Otro ejemplo sería definir dos métricas instanciadas: “Edad del usuario” y “Teléfono de usuario”. La métrica instanciada “Edad del usuario”, de granularidad Conjunto de Celdas, determina si la edad del usuario es válida en base a la fecha de nacimiento. La métrica instanciada “Teléfono de usuario” válida si el teléfono es de largo 9, la granularidad es celda. Nótese que las métricas instanciadas definidas tienen granularidades compatibles.
Se define una métrica instanciada compuesta que determina si los valores son verdaderos. La métrica instanciada compuesta aplicada a ambas métricas instancias retorna un resultado por cada registro. Es decir, determina si el teléfono y la edad de cada usuario son válidos.

Para brindar mayor flexibilidad y optimizar los recursos disponibles (poder de cómputo y espacio en disco), al momento de ejecutar una métrica instanciada compuesta, se puede determinar si se ejecutan las métricas instanciadas que la forman o se toman los resultados de ejecuciones previas. También se tiene que poder determinar qué resultados se van a guardar en la base de metadatos de calidad en el caso de ejecutar las métricas instanciadas compuestas. Las posibilidades son guardar todos los resultados (los de las métricas instanciadas y los de la métrica instanciada compuesta) o solo los de la métrica compuesta.

iv. Requerimientos funcionales

A continuación, se detallarán los requerimientos funcionales enumerados anteriormente.
La herramienta consta de tres módulos: catálogo, modelo de calidad y ejecución. En la sección catálogo se configurarán las dimensiones, los factores, las métricas y los métodos quedando disponibles para utilizar en los modelos de calidad. En la sección modelo de calidad se podrán definir las métricas instanciadas utilizando los elementos del catálogo, las métricas instanciadas compuestas y las agregaciones. En la sección ejecución se ejecutan las métricas instanciadas, las agregaciones y las métricas instanciadas compuestas y se visualizan los resultados.

1. Requerimientos de configuración

En esta sección vamos a describir los requerimientos de configuración de la herramienta para definir los componentes del modelo de calidad. Para todos los casos se asume que el usuario que realiza la acción cuenta con los permisos necesarios.

- ABM dimensiones
  Se podrá dar de alta, modificar y eliminar dimensiones. Para definir una dimensión es necesario especificar el nombre de esta. No van a poder existir dos dimensiones con el mismo nombre y no se puede eliminar una dimensión si tiene factores asociados.
- ABM factores
  Se darán de alta, modificarán y eliminarán factores. Para definir un factor es necesario especificar el nombre y seleccionar la dimensión a la que pertenece. El nombre del factor es único y no se va a poder eliminar un factor que tenga métricas asociadas.
- ABM métricas
  Para definir una métrica es necesario especificar el nombre y seleccionar la dimensión, el factor y la granularidad. El nombre de la métrica será único y no se puede eliminar una métrica si tiene algún método asociado. Se podrá modificar una métrica.
- ABM métodos
  Para dar de alta un método se debe seleccionar la dimensión, el factor y la métrica que implementa y especificar el nombre del método, la URL y la operación del WS que contiene la implementación. El nombre del método es único. Los métodos se podrán modificar y eliminar, en el caso que no se utilice en ningún modelo de calidad.

2. Requerimientos del modelo de calidad

A continuación, detallamos los requerimientos para definir y trabajar con un modelo de calidad.

- Definir base de datos
  Para definir la base de datos a evaluar se van a ingresar las credenciales necesarias para que se establezca la conexión, también se debe seleccionar en qué motor de base de datos esta dicha base.
- ABM métricas instanciadas
  Se podrán dar de alta, modificar y eliminar métricas instanciadas. Para dar de alta se debe seleccionar la dimensión, el factor, la métrica, el método, los elementos a evaluar de la base de datos. Solo se podrá modificar el nombre y no se podrá eliminar cuando está asignada a una agregación.
- ABM agregaciones
  Se podrá dar de alta, modificar y eliminar agregaciones. La agregación consta de un nombre, una métrica instanciada, la granularidad final, la URL y la operación del WS.
- ABM métricas instanciadas compuestas
  Se podrán dar de alta, modificar y eliminar métricas instanciadas compuestas. Para dar de alta se debe seleccionar las métricas instanciadas que la conforman, la URL y la operación del WS. Las restricciones son que las métricas instanciadas que lo forman tienen que ser de granularidades compatibles y tienen que estar aplicadas a los mismos datos. Esto es importante para asegurar que los resultados son combinables.
- Visualizar modelo de calidad
  Se podrá visualizar el modelo de calidad incluyendo las dimensiones, los factores, las métricas, los métodos, las métricas instanciadas, las agregaciones definidas y las métricas instanciadas compuestas.
- Exportar e imprimir modelo de calidad
  Se podrán exportar a un archivo y/o imprimir los modelos de calidad. Para cada modelo de calidad se incluirán las dimensiones, los factores, las métricas, los métodos, las métricas instanciadas, y las agregaciones y las métricas instanciadas compuestas que se definieron.

3. Requerimientos de la ejecución de la medición

Detallamos los requerimientos para ejecutar las métricas instanciadas, las agregaciones y las métricas instanciadas compuestas.

- Base de metadatos de calidad
  El sistema generará las bases de metadatos de calidad para almacenar los datos utilizados en las evaluaciones de la calidad y el resultado obtenido luego de la ejecución.
- Ejecutar modelo de calidad
  Una vez definido el modelo de calidad, tiene que ser posible ejecutarlo completo o un subconjunto de métricas instanciadas seleccionadas. Una vez ejecutado se mostrará en pantalla los resultados de la ejecución. Los resultados de la última ejecución se guardarán en la base de metadatos de calidad, haciendo referencia al dato evaluado.

- Exportar e imprimir resultados
  Los resultados de la ejecución del modelo de calidad se mostrarán en un reporte que se podrá exportar e imprimir.

v. Requerimientos no funcionales

- Debe ser web.
- Tiene que estar implementada en Java.
- Las bases de datos a evaluar deben ser relacionales, al menos MySQL y PostgreSQL.
- La interfaz debe estar en inglés.

vi. Modelo de dominio

Con el objetivo de hacer un resumen de los conceptos que vamos a modelar y la relación entre ellos, en la siguiente sección vemos el modelo de dominio y las restricciones.

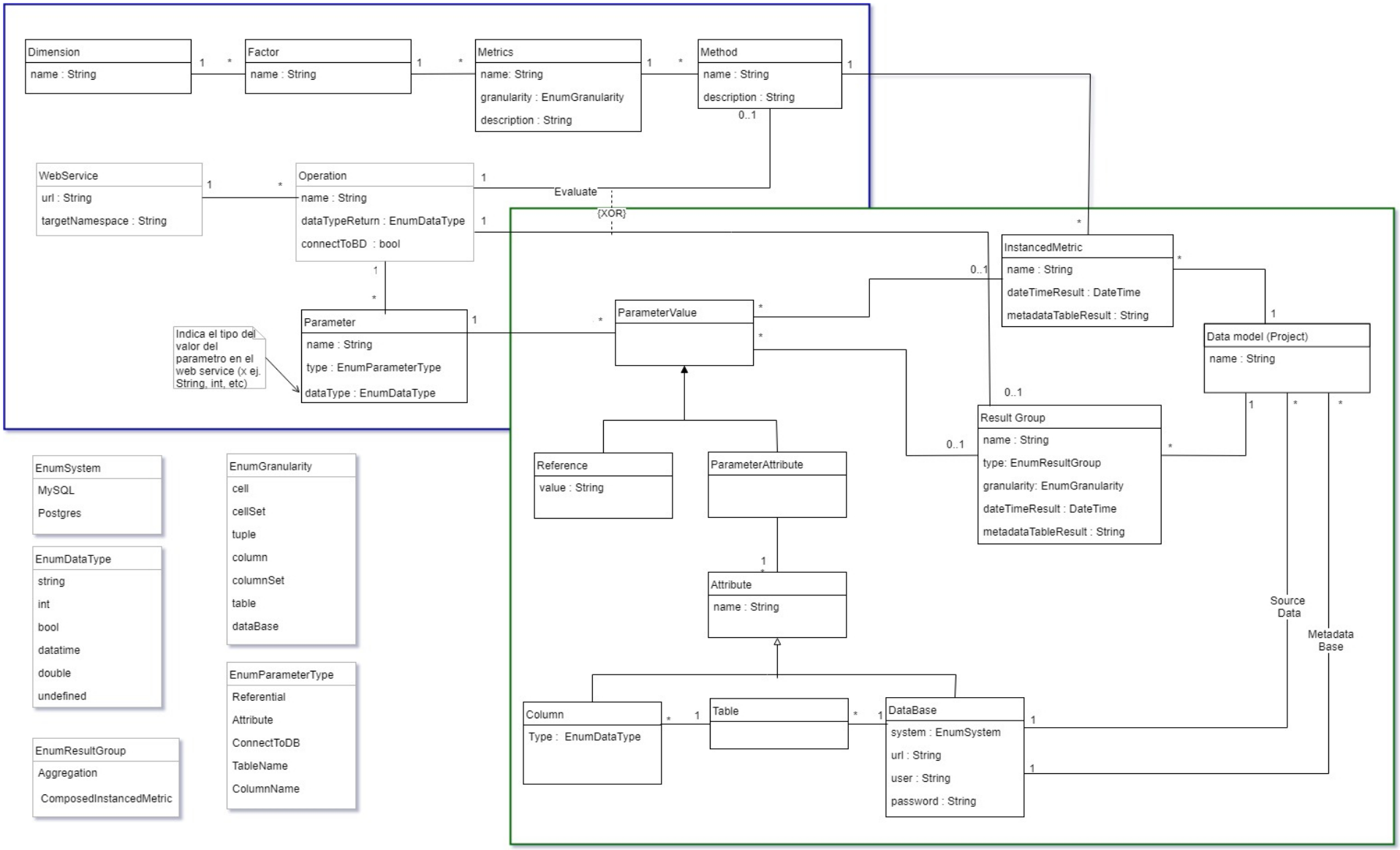


Figura 6: Modelo de dominio

Como vemos en la figura 6, el modelo de dominio lo podemos dividir en dos submodelos, por un lado (azul) modelamos la configuración de la herramienta, es decir, dimensiones/factores/métricas y métodos que se van a poder utilizar. El nombre identifica las entidades dimensión, factor, métrica y método.

Por otro lado (verde) modelamos la definición de un modelo de calidad de datos para una base y la forma de guardar los resultados de sus ejecuciones, utilizando los conceptos definidos en la configuración. El modelo de datos está compuesto por las métricas instanciadas y los resultados de grupo. Los resultados de grupo son las agregaciones y métricas instanciadas compuestas.
Las métricas instanciadas van a definir qué método le vamos a aplicar al dato que queremos analizar y cómo se mapean los atributos de la base de datos a analizar con los parámetros del web service que implementan los métodos. Es la entidad “ParameterValue” la encargada de realizar dicho mapeo. Es preciso aclarar que los parámetros de una métrica instanciada deben coincidir con los parámetros del método que la implementa. Las métricas instanciadas van a tener dos tipos de parámetros asociados, los parámetros referenciales, es decir, valores fijos y los parámetros que hacen referencia a los datos a evaluar. Estos últimos los modela la entidad “ParameterAttribute”. Para las métricas instanciadas definidas para una base de datos, los parámetros del tipo atributo deben ser atributos (columna, tabla o base de datos) de dicha base de datos.
Los resultados de grupo son las agregaciones y métricas instanciadas compuestas. Para su definición los posibles parámetros son referenciales o métricas instanciadas de su mismo modelo de datos, no son atributos de la base de datos. Al igual que para las métricas instanciadas la entidad “ParameterValue” mapea los parámetros del web service que implementa el resultado de grupo con los valores de la base de metadatos a invocar.
Cuando es del tipo agregación sólo puede tener asociada una métrica instanciada, si es del tipo métrica instanciada compuesta tiene asociada al menos dos métricas instanciadas.

Para finalizar la etapa de análisis en el Anexo I: Casos de Uso especificamos los casos de uso de las principales funcionalidades.

4.1. Diseño

A continuación, se presentan las definiciones de diseño que fuimos tomando una vez que tuvimos los requerimientos definidos, tanto los funcionales como los no funcionales. Además, para algunos requerimientos nos planteamos varios escenarios de posibles soluciones, de los cuales discutimos las ventajas y desventajas. En esta sección documentamos dicha discusión.

4.1.1. Métodos de medición

Dentro de los requerimientos no funcionales del sistema hay algunos que consideramos claves para tomar algunas de las decisiones de diseño. Uno de estos requerimientos es la necesidad de que la herramienta sea flexible y extensible, es decir, que no esté limitada a una cierta cantidad de operaciones o que tenga definido de una forma rígida las dimensiones/factores/métricas y métodos que se pueden utilizar para evaluar los datos.

Tiene que ser posible adaptar la herramienta a nuevas clasificaciones y poder extender las operaciones que implementan los métodos definidos según cambien las necesidades y sin tener que re-implementarla. Para esto la herramienta cuenta con un módulo de configuración, dicho módulo tiene un ABM de dimensiones/factores/métricas y métodos de medición. Para poder cumplir con los requerimientos nos queda definir cómo implementar los métodos de medición de una forma que permita, de forma sencilla, agregar nuevos. Para esto lo que vamos a utilizar son web services. Cada método será definido por una operación de un web service y este web service es el encargado del procesamiento de los datos y generar la medida de calidad.

En este punto tenemos que definir cómo vamos a pasarle al WS los datos que queremos medir. Cuando la granularidad de la métrica es celda, conjunto de celda o tupla, es natural pensar que lo podemos pasar como parámetros del WS, el problema surge cuando la granularidad es columna, tabla o base de datos. Ya que en estos casos se dificulta pasar toda la información como parámetros del WS. Para definir la interacción entre los web services y la herramienta, planteamos tres casos posibles y analizamos la viabilidad de cada uno:

- Caso 1: La herramienta consulta la base de datos a evaluar, luego invoca el web service con los valores a ser evaluados y por último guarda el resultado en la base de metadatos de calidad, en la figura 7 muestra la interacción entre la herramienta, los WS y las bases de datos.

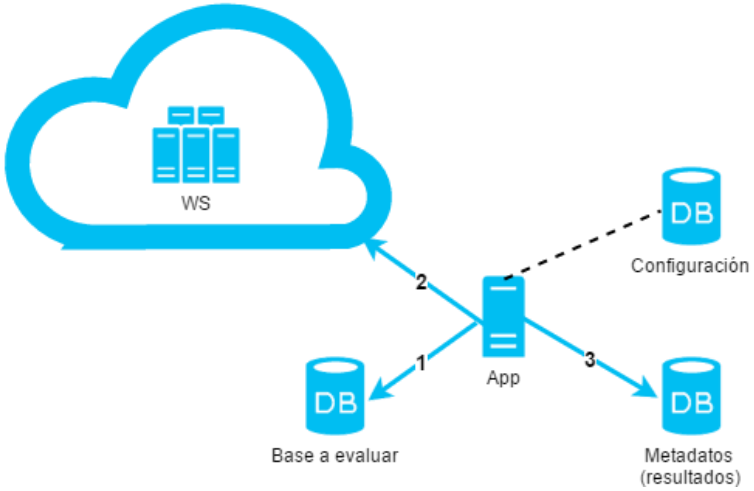


Figura 7: Interacción WS/herramienta - Caso1

Para las granularidades de tipo celda, conjunto de celdas y tupla no encontramos una clara desventaja a esta solución. Sin embargo, para las otras granularidades nos encontramos con el problema de invocar al web service con muchos datos, por ejemplo, una columna con un millón de registros. Por lo tanto, adoptamos esta solución para las granularidades celda, conjunto de celda y tupla y continuamos el estudio para las otras granularidades.

- Caso 2: La herramienta invoca al web service con los datos de conexión a las dos bases de datos (la que contiene los datos a evaluar y la base de metadatos de calidad). De esta forma es el web service el encargado de consultar el dato, evaluarlo y guardar el resultado.

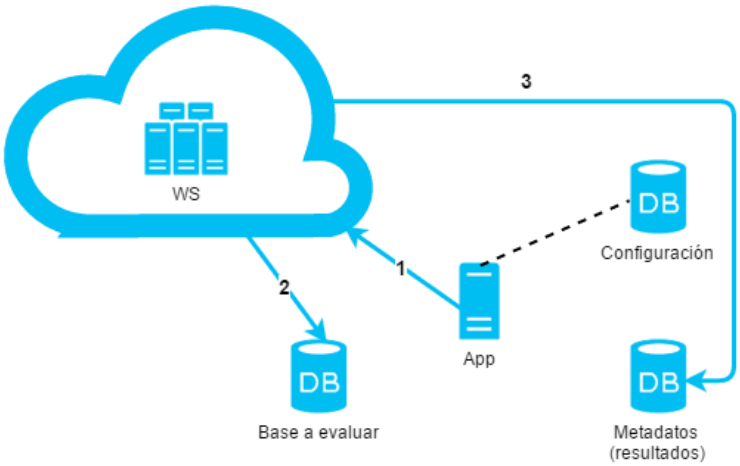


Figura 8: Interacción WS/herramienta - Caso 2

En este caso resolvemos el problema que teníamos para las granularidades columna, conjunto de columnas, tabla y base de datos con respecto al volumen de datos con el que invocamos al web service. La desventaja, es que la implementación del web service tiene que conocer en detalle la base de metadatos de calidad para poder guardar los resultados de manera correcta y con las referencias adecuadas. Le quita flexibilidad a la solución de implementar los métodos vía web services y la posibilidad de reutilización de estos.

- Caso 3: La herramienta invoca al web service con los datos de conexión a la base de datos a evaluar, el web service se encarga de consultar los datos y devuelve un resultado. Es la herramienta la encargada de guardar el resultado en la base de datos de metadatos.

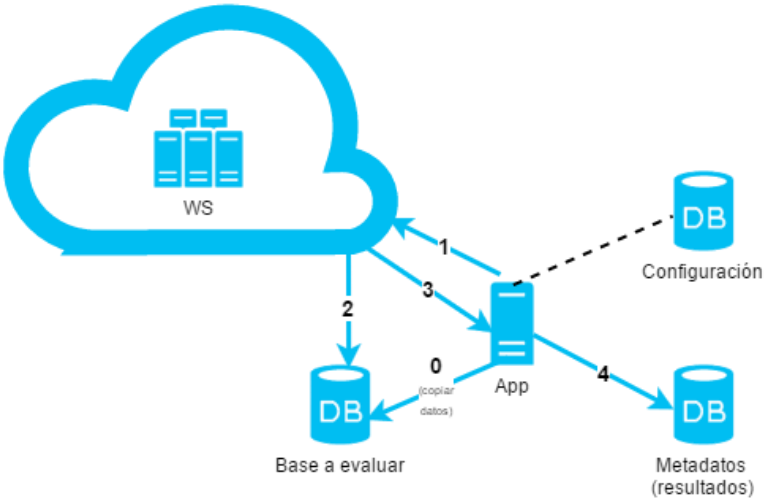


Figura 9: Interacción WS/herramienta - Caso 3

A diferencia del caso 2, en este caso es la herramienta la que guarda los resultados. Por lo tanto, para las granularidades que teníamos pendiente resolver, esta solución resuelve las desventajas que planteamos en los casos anteriores.

En resumen, serán dos tipos de web services: los que reciben los datos a evaluar de la base de datos (celda, conjunto de celda y tupla) y los que tienen como entrada el string de conexión

a la base de datos para que la herramienta se conecte directamente a dicha base para obtener los datos a evaluar (columna, tabla y base de datos). Además, en este caso es necesario que la operación reciba como entrada los nombres de los elementos de la base de datos a los que se les va a evaluar la calidad y la URL de conexión de dicha base. Una vez que el modelo de calidad se ejecute, el resultado será guardado en la base de metadatos.

Algunos métodos necesitarán otras entradas, como por ejemplo valores de referencia. Los valores de referencia son los valores necesarios para ejecutar el método, por ejemplo, un método para la dimensión consistencia y el factor integridad de dominio podrá recibir como entrada la especificación del dominio. En el caso de la integridad de la edad, recibirá como información de referencia el rango de edad válido.

Las métricas instanciadas se configurarán como se explicó anteriormente, utilizando los dos tipos de web service.
Las agregaciones se configurarán de forma que el web service se conecte a la base de metadatos para obtener los datos a evaluar.
Las métricas instanciadas compuestas utilizarán web services que recibirán en cada parámetro de la operación el resultado de la ejecución de una métrica instanciada.

Tanto para el caso de los web services que implementan los métodos que evalúan la calidad, como los web services que realizan agregaciones y métricas instanciadas compuestas, dentro del alcance de este proyecto está la implementación de un conjunto básico de los mismos. Además, se pueden utilizar web services existentes.

4.1.2. Metadatos

Otro aspecto importante que tuvimos en cuenta en la etapa de diseño es cómo vamos a guardar los resultados luego de ejecutar el modelo de calidad definido. En este punto las dificultades están en tener referencias a los datos evaluados, es decir, poder asegurar la trazabilidad entre el resultado y el dato que fue evaluado, sin excedernos en la cantidad de datos que copiamos o duplicamos. En este punto evaluamos varias opciones:

A. Copiar toda la base de datos.
   Antes de ejecutar el modelo de calidad copiamos la base de datos a evaluar entera. Con esta solución podemos garantizar la trazabilidad entre el dato analizado y el resultado. Además, en el caso que se definan nuevas métricas para dicho modelo, podemos volver a ejecutarlo sobre la misma versión de los datos que el anterior. Tampoco necesitamos que los datos a evaluar tengan definida una clave primaria, ya que no vamos a hacer referencia. El problema de esta solución es que si la base de datos es grande el espacio/tiempo que ocupa/demora en realizar la copia puede ser grande y no siempre son necesarios todos los datos, depende lo que analice el modelo definido.
B. Copiar datos a ser evaluados.
   Al momento de ejecutar el modelo de calidad copiamos los datos que van a ser evaluados, no toda la base de datos. La ventaja de esta opción es que no estamos copiando toda la base de datos, por lo tanto, si la base es grande, el tiempo/espacio que demora/ocupa va a ser proporcional a lo que se quiere evaluar. Ante un cambio en el modelo de calidad que incluya datos nuevos, es necesario volver a copiar los datos anteriores para asegurar que el modelo de calidad evalúa la misma versión de la base de datos para todos los datos a analizar.

C. Referenciar datos
   No copiamos los datos a la base de metadatos, tomamos el dato de la base original, lo analizamos y guardamos el resultado del análisis junto con la referencia al dato. En esta opción nos olvidamos del tamaño de la base de datos a evaluar, pero nos encontramos con al menos dos inconvenientes, el primero es que tenemos que asegurarnos que la base a evaluar tiene que tener bien definidas las claves primarias para poder ser referenciadas. Y el segundo es que al consultar la medición de un dato no podemos asegurar que el valor actual del mismo sea el que se analizó.
D. Sin referenciar.
   Guardamos únicamente los resultados del análisis sin copiar, ni referenciar los datos analizados. Al igual que en la opción anterior nos deja de preocupar el tamaño de la base de datos a evaluar. Igualmente, no es una opción que nos interese para la herramienta ya que es importante poder referenciar el dato analizado.

Luego de analizar estas opciones, con las ventajas y desventajas que encontramos y teniendo en cuenta los requerimientos que tiene la herramienta que vamos a construir llegamos a las siguientes conclusiones:

- Para asegurar la referencia al dato analizado, garantizando el valor al momento del análisis, decidimos copiar los datos a ser analizados a la base de metadatos. No se va a realizar la copia de la base de datos entera, sino sólo de aquellos datos que participan de las métricas instanciadas definidas para el modelo de calidad que se va a ejecutar (Opción B).
- Al ejecutar el modelo de calidad si tenemos copiados los datos que se van a analizar el usuario debe elegir si ejecuta el modelo con dichos datos o si realiza una nueva copia.
- Los resultados de la ejecución del modelo de calidad se guardarán para la última ejecución, para las ejecuciones anteriores se guardará un resumen. De esta forma el usuario podrá monitorear a lo largo del tiempo la calidad de los datos, pero sin poder entrar en detalle profundo. Para este resumen se utilizarán las agregaciones definidas para el modelo y en el caso que no haya definidas habrá una por defecto.

En base a estas definiciones, modelamos la base de metadatos de calidad, a continuación, detallamos el diseño y el proceso de creación de dicha base.

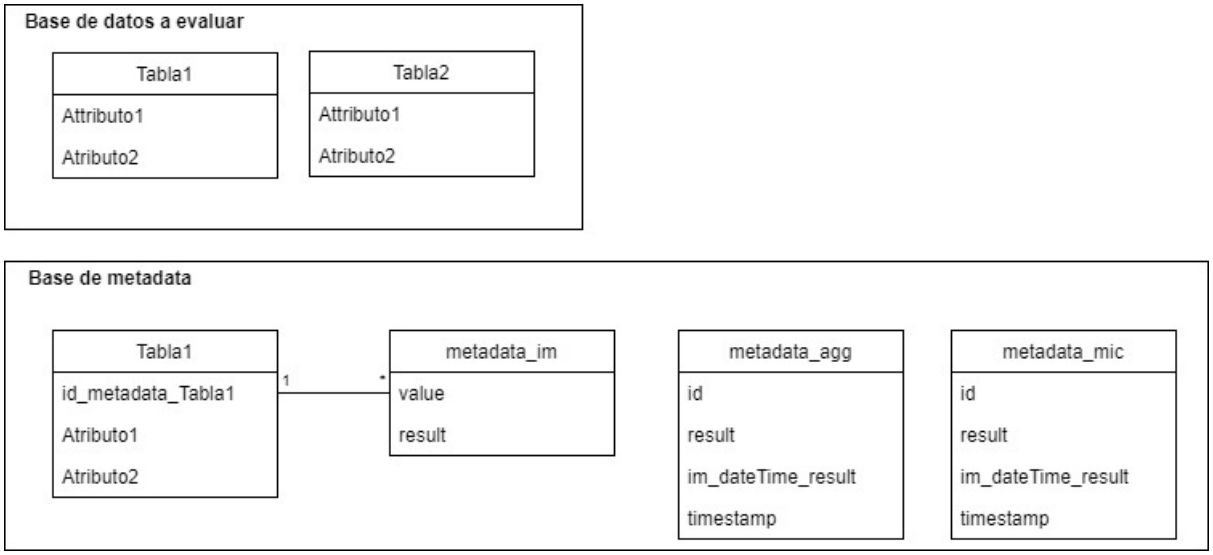


Figura 10: Diseño de la base de metadatos

Antes de evaluar una métrica instanciada se copian los datos que participan de las métricas que se van a ejecutar a la base de metadatos, agregando un identificador a cada registro. Este identificar lo agregamos para asegurarnos que los datos tengan una clave primaria y se pueda referenciar.
Luego de ejecutar la métrica instanciada se creará la tabla llamada metadata_im. Para las granularidades celda, conjunto de celdas y tupla, dicha tabla estará compuesta por el resultado de la ejecución de la métrica instanciada y una referencia al dato en la base de metadatos. Este dato es el que copiamos previo a la ejecución y lo referenciamos a través de la clave primaria que agregamos. Para las restantes granularidades se guarda solamente el resultado. El diseño de la base de metadatos se muestra en la figura 10.

Para las agregaciones se crea una tabla por cada agregación y luego de la ejecución se registra el resultado, con la fecha de ejecución de la métrica instanciada y timestamp.

En el caso de las métricas instanciadas compuestas se crea una tabla metadata_mic para cada una, siendo el diseño igual al de la de agregaciones.

En los nombres de las tablas metadata_im, metadata_agg y metadata_mic se agrega el identificador a la entidad que corresponde. Estos nombres son referenciados en la base de datos de configuración junto a la información de la métrica instanciada, agregación y métrica instanciada compuesta respectivamente.

4.2. Arquitectura

En esta sección presentamos el diagrama de los componentes del sistema y la interacción entre cada uno de ellos, es decir, el diagrama de la arquitectura del sistema (figura 11). Además, comentamos las responsabilidades de cada uno de ellos.

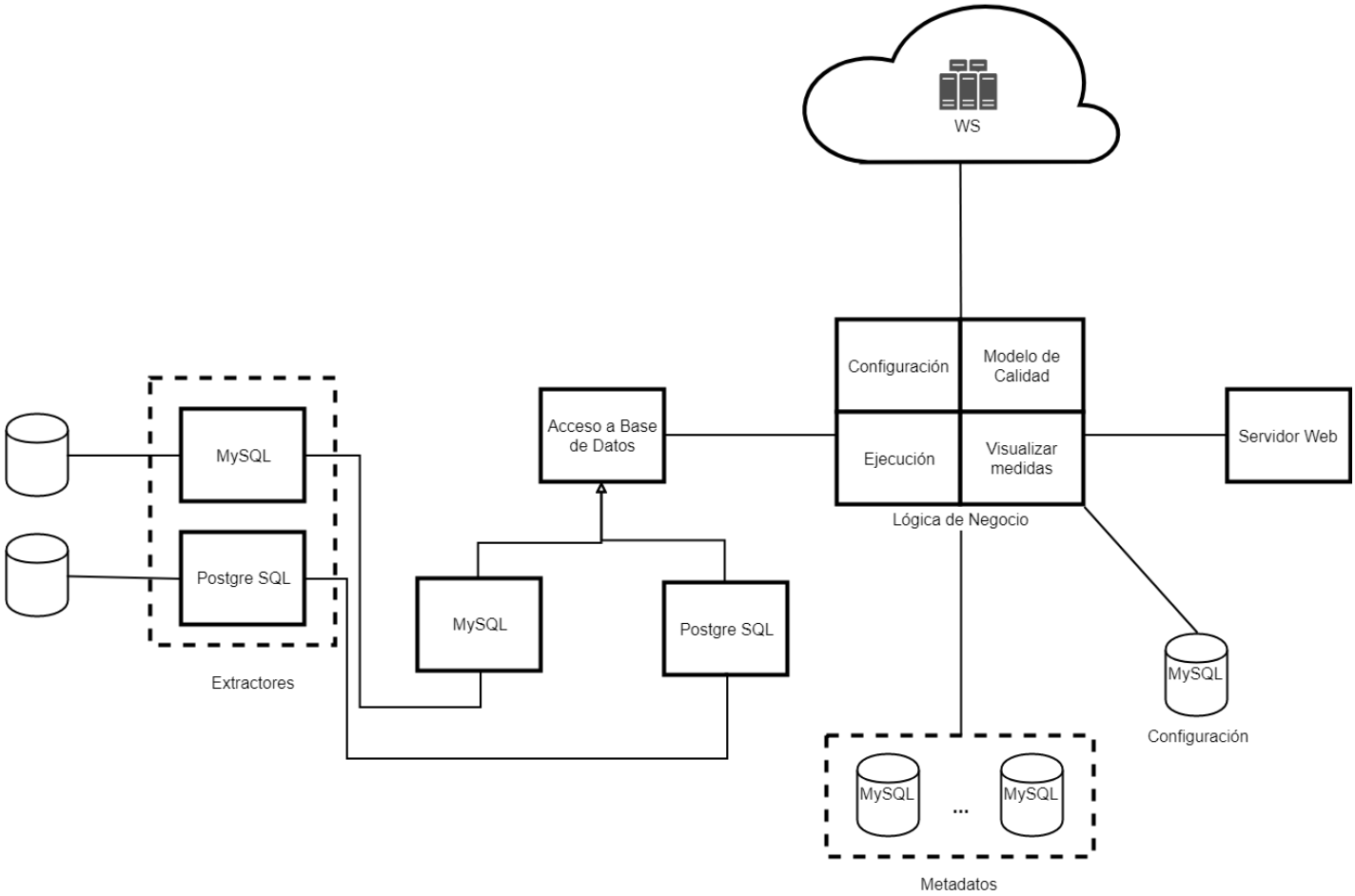


Figura 11: Arquitectura del sistema

El sistema se divide en dos grandes componentes “Lógica de negocio” y “Acceso a base de datos”.
La “Lógica de negocio” se subdivide en los módulos “Configuración”, “Modelo de calidad”, “Ejecución” y “Visualizar medidas”.

El módulo “Configuración” se encarga del alta, baja y modificación de las dimensiones, los factores, las métricas y los métodos.
En el módulo “Modelo de calidad” se dan de alta, baja y modifican las métricas instanciadas, las agregaciones y las métricas instanciadas compuestas. Además, se visualiza la definición del modelo.
El módulo “Ejecución” preprocesa los datos, invoca los web services y post-procesa los resultados de las ejecuciones de las métricas instanciadas, agregaciones y métricas instanciadas compuestas. También se encarga de la visualización de los resultados de las ejecuciones.

La configuración y el modelo de calidad se almacenan en una base de datos MySQL, la cual llamaremos base de configuración.
Cada módulo de la “Lógica de negocio” se comunica con la base de configuración.

El “Acceso a base de datos” se encarga de obtener los datos a evaluar. Se subdivide en dos módulos “MySQL” y “PostgreSQL”, cada uno contiene la lógica necesaria para obtener y guardar los datos de las bases que correspondan.

Las bases de metadatos contienen los datos analizados y sus resultados, cada base analizada va a tener su propia base de metadatos de calidad. La base de metadatos de calidad siempre es MySQL.

5. Implementación

Este capítulo está dedicado a la implementación de la herramienta, primero repasamos las herramientas que utilizamos en el desarrollo, luego recorremos los problemas técnicos que fuimos enfrentando y las soluciones que encontramos. Además, repasamos como implementamos las métricas instanciadas, agregaciones y las operaciones que incluimos en el catálogo. Incluimos también una guía de cómo se configura la herramienta y por último mencionamos algunas claves para extender el proyecto.

5.1. Herramientas utilizadas

El desarrollo de la solución se realizó en el entorno Eclipse Java EE IDE for Web Developers [19]. Las dependencias y componentes externos se gestionaron con Apache Maven [20].
La información de configuración y de los modelos de datos se almacenó en una base de datos MySQL [23]. Como herramienta ORM utilizamos Hibernate [26].
En el desarrollo de la web se utilizó el framework Spring Web MVC versión 4.3.1 [21] y Bootstrap 3.3.7 [22] para el diseño del frontend. La exportación a PDF se realizó con iText [27].
Los web services se manipularon con predic8 [28].
El manejo de control de versiones del código se realizó con Git [24] y se respaldó en Bitbucket [25].

5.2. Problemas y soluciones

En la etapa de desarrollo nos fuimos enfrentando a diferentes problemas que requirieron analizar posibles soluciones, teniendo en cuenta los pros y contras de cada una, fue necesario tomar algunas decisiones. A continuación, vamos a hacer un repaso de dichas decisiones.

Cada motor de bases de datos tiene un conjunto diferente de tipos de datos, los cuales no se corresponden directamente a los tipos de datos que brinda java. En la documentación de MySQL [31] proponen mapearlos a los tipos numéricos y string de java. Para dar más flexibilidad a la implementación de los web services soportamos los tipos string, integer, bool, datetime y double de java. En la tabla 3 se visualizan como realizamos los mapeos de los tipos entre java, MySQL y PostgreSQL.

Java	**MySQL [32]**	**PostgreSQL [33]**
String	char, varchar, binary, varbinary, blob, text, enum, set, tinyblob, mediumblob, longblob, tinytext, mediumtext, longtext, enum, set, json	character, character varying, cidr, inet, json, macaddr, text, tsquery, tsvector, uuid, xml
Integer	tinyint, smallint, mediumint, int, integer, bigint, bit(n), year	bit, bit varying, bigint, bigserial, bytea, character, integer, smallint, smallserial, serial

Bool	bit(1)	boolean
Datetime	date, time, datetime, timespan	date, interval, time, timestamp
Double	decimal, float, double, numeric, real	double precision, money, numeric, real

Tabla 3: Mapeo de tipos de datos de Java a MySQL y PostgreSQL

Los tipos de datos espaciales de MySQL y los geométricos de PostgreSQL no se incluyeron en la implementación, ya que excedían el alcance del proyecto.

El tipo de datos fecha y hora puede tener diferentes formatos dependiendo de la configuración regional de la computadora. Para no limitar la aplicación a determinados formatos se decidió que en el caso que los datos en base de datos sean fechas y horas el parámetro en el web service puede ser del tipo String, Date o Datetime. En el caso que el parámetro sea del tipo String en el web service se puede implementar un XmlAdapter para convertirlo al tipo Date o Datetime que corresponda.

Las bases de metadatos son creadas por la aplicación para cada modelo de calidad, el administrador de la aplicación puede configurar la ubicación de dichas instancias. La aplicación creará los esquemas necesarios y la base de datos, el nombre de la base de datos no se podrá modificar.

Como medida de seguridad la contraseña de las bases de datos fuente se almacenan cifrados.

Al ejecutar el modelo de calidad la transferencia de datos, su obtención desde la base de datos y/o el procesamiento, cuando son grandes cantidades, demoran un tiempo considerable, afectando el funcionamiento de la aplicación. Por lo tanto, fue necesario que cada modelo de calidad se ejecute en un hilo distinto.

5.3. Métricas instanciadas, métricas instanciadas compuestas y agregaciones implementadas

En esta sección del informe repasamos como implementamos los conceptos de métricas instanciadas, métricas instanciadas compuestas y agregaciones. Además, enumeramos las métricas y agregaciones que incluimos en catálogo de la herramienta.

5.3.1. Implementación

En la etapa de diseño decidimos implementar las métricas a través de web services para cumplir con el requerimiento de extensibilidad de la herramienta.

Cada método, que implementa una métrica instanciada, es una operación de un web service. Para dar de alta un método en el catálogo, además de indicar la métrica instanciada que implementa, es necesario indicar la URL del web service que la va a implementar. La herramienta valida la URL ingresada y lista las operaciones disponibles. Al seleccionar la

operación a utilizar, se listan los parámetros y se debe indicar el tipo de cada parámetro. Lo implementamos de esta forma con el fin de reutilizar métodos y no tener que crear operaciones para métodos similares. Por ejemplo, los teléfonos fijos de Uruguay tienen 8 caracteres y los códigos postales de Alemania tienen 5 caracteres. Por lo tanto, si implementamos una operación que recibe solamente el parámetro a evaluar, para cubrir los dos casos tenemos que implementar dos operaciones, uno para comparar el largo con 8 y otro con 5. Pero si implementamos una operación que recibe el dato a evaluar y un parámetro referencial que vamos a utilizar para comparar, con una sola operación podemos cubrir varios casos. Dependiendo de la granularidad de la métrica, son las opciones de configuración disponibles para cada parámetro, esto se debe a la interacción que va a tener el web service con los datos. Para las granularidades celda, conjunto de celdas y tupla, los tipos de parámetro son:

- Atributo: es un tipo de parámetro que va a recibir el dato a evaluar.
- Referencial: es un tipo de parámetro que va a recibir un valor fijo.

Para las granularidades columna, conjunto de columnas, tabla y base de datos, los tipos de parámetros son:

- Parámetro de conexión: este tipo de parámetros va a recibir el string de conexión a la base de datos a evaluar.
- Nombre de columna: este tipo de parámetro recibe el nombre de una columna.
- Nombre de tabla: es un tipo de parámetro que recibe el nombre de una tabla.
- Referencial: como en el caso anterior, es un tipo de parámetro que va a recibir un valor fijo.

Al momento de dar de alta los métodos en el catálogo se realiza la configuración que detallamos, luego al momento de instanciar las métricas, se define los valores con los que se va a invocar el web service. Por ejemplo, para los parámetros del tipo referencial, al momento de definir la métrica instanciada se tiene que definir el valor que va a tener dicho parámetro.

5.3.2. Catálogo

A continuación, detallamos las operaciones que incluimos en el catálogo haciendo una breve descripción de cada una de ellas y especificando la dimensión, factor, métrica y granularidad asociadas a dicho método.

Método 1: Verificar el dígito verificador de las CI uruguayas
Dimensión: Exactitud
Factor: Exactitud sintáctica
Métrica: Booleano de exactitud sintáctica
Granularidad: Celda
Descripción: Recibe como parámetro un string que representa un número de cédula uruguaya, se verifica si el dígito verificador es correcto y devuelve 1 en caso de que sea válido y si no 0.

Método 2: Verificar si un string cumple con una expresión regular
Dimensión: Exactitud
Factor: Exactitud sintáctica
Métrica: Booleano de exactitud sintáctica
Granularidad: Celda

Descripción: Esta operación recibe dos parámetros, uno va a ser del tipo atributo en donde va a recibir el dato a evaluar y el otro va a ser del tipo referencial en donde va a recibir el string de la expresión regular el cual se va a utilizar para realizar la validación. Esta operación va a devolver 1 en caso de que verifique la expresión regular y 0 si no la verifica.

Método 3: Verificar el largo de un string
Dimensión: Exactitud
Factor: Exactitud sintáctica
Métrica: Booleano de exactitud sintáctica
Granularidad: Celda
Descripción: Como el caso anterior, recibe dos parámetros uno del tipo atributo en donde recibimos el string al que se le quiere verificar el largo y uno del tipo referencial que es donde se recibe el largo que tiene que tener. Devuelve 1 en caso de que el largo del string sea igual al especificado y 0 si no.

Método 4: Porcentaje de valores no nulos
Dimensión: Completitud
Factor: Densidad
Métrica: Ratio de densidad
Granularidad: Columna
Descripción: Recibe como parámetros el string de conexión a la base de datos, el nombre de la columna y de la tabla de los datos que se quiere evaluar. Devuelve el porcentaje de valores no nulos de la columna.

Método 5: Porcentaje ponderado de valores no nulos
Dimensión: Completitud
Factor: Densidad
Métrica: Ratio de densidad
Granularidad: Conjunto de celdas
Descripción: Esta operación recibe cuatro parámetros del tipo atributo y cuatro parámetros del tipo referenciales que van a definir el peso de cada parámetro en el promedio ponderado. Devuelve el porcentaje ponderado de valores no nulos. Un ejemplo de aplicación de esta operación es si queremos evaluar la completitud de una dirección que guardamos como (calle, esquina, número, ciudad y para considerar la dirección “más completa” la calle tiene un peso del 50%, la esquina y número el 20% y la ciudad un 10%.

Método 6: Verificar si un número se mantiene por debajo del valor máximo definido
Dimensión: Consistencia
Factor: Integridad de dominio
Métrica: Booleano de Consistencia
Granularidad: Celda
Descripción: Recibe como parámetro del tipo atributo un entero y como parámetro referencial el máximo valor que puede tomar. Devuelve verdadero si el entero es menor al de referencia, en otro caso falso. Esta operación sirve por ejemplo para verificar si la edad se encuentra por debajo de los 120 años.

Método 7: Verificar si un entero es menor a un número dado y otro entero es mayor a otro número dado

Dimensión: Consistencia
Factor: Integridad intra-relación
Métrica: Booleano de Consistencia
Granularidad: Conjunto de celdas
Descripción: Recibe 2 parámetros del tipo atributo enteros y dos parámetros del tipo referencial. Compara si el primer parámetro entero es menor al primer parámetro referencial y si el segundo parámetro entero es mayor al otro parámetro referencial. Devuelve 1 si se cumplen las dos condiciones, en otro caso 0.

Método 8: Verificar si la edad es correcta dada la fecha de nacimiento
Dimensión: Consistencia
Factor: Integridad intra-relación
Métrica: Booleano de Consistencia
Granularidad: Conjunto de celdas
Descripción: Recibe dos parámetros del tipo atributo, la edad y la fecha de nacimiento, realiza la cuenta y si es correcta devuelve 1, sino 0.

Método 9: Porcentaje de no duplicados
Dimensión: Unicidad
Factor: No-duplicación
Métrica: Ratio de no-duplicados
Granularidad: Columna
Descripción: Recibe como parámetros el string de conexión a la base de datos, los nombres de la columna y de la tabla de los datos que se quiere evaluar. Devuelve el porcentaje de valores no duplicados.

Para el caso las agregaciones y de las métricas instanciadas compuestas implementamos web services que reciben dos valores, los combina y devuelve un resultado. Las operaciones de combinación que incluimos en el catálogo son:

- Suma
- Promedio
- OR
- AND

5.4. Configuración de la herramienta

En esta sección explicaremos las configuraciones iniciales de la herramienta, también repasamos cómo extenderla a otros motores de bases de datos relacionales. Las bases de datos no-relacionales exceden el alcance del proyecto.

En el archivo kiuplus.properties se realiza la configuración inicial del proyecto, los parámetros a definir son:

- metadata.database.password=admin
- metadata.database.name=kiuplus_metadatos
- metadata.database.username=root
- metadata.database.url=jdbc\:mysql\://localhost\:3306

- images.source=http\://localhost\:8080/kiuplus/resources/images

Como se definió anteriormente el motor de base de datos que utiliza la herramienta para guardar la información generada es MySQL. Los parámetros “metadata.database.username” y “metadata.database.password” son el usuario y contraseña para establecer la conexión a la base de datos. Es necesario que el usuario tenga permisos de lectura, escritura, creación y eliminación de tablas y bases de datos.
La URL de conexión a la base de datos se configura en “Metadata.database.url” y el nombre de la base de datos en “metadata.database.name”. Esta base de datos es creada por KiuPlus y el nombre lo puede elegir el administrador.
Las imágenes de KiuPlus se encuentran en la URL indicada en “images.source”, en este parámetro es necesario cambiar el host y port de la aplicación, quedando el resto de la URL igual.
Es necesario configurar como variable de entorno del TomCat el archivo kiuplus.properties. El nombre de la variable es “CONF_DIR” y el valor es la ruta absoluta al archivo, por ejemplo: “C:\KiuPlus”.

Si bien la herramienta permite definir las dimensiones, factores, métricas y métodos que el usuario quiera, como parte de la configuración inicial se incluyen en el catálogo las dimensiones y los factores listadas en la sección Dimensiones de calidad, las métricas definidas en [5] y los métodos definidos en Métricas instanciadas, métricas instanciadas compuestas y agregaciones implementadas. A modo de resumen las listamos en la siguiente tabla.

Dimensión	**Factor**	**Métrica**	**Método**
Accuracy	Semantic correctness	Sem. corr. boolean	
		Sem. corr. degree	
		Sem. corr. deviation	
	Syntactic correctness	Synt. corr. boolean	**Método 1:** Verificar el dígito verificador de las CI uruguayas **Método 2:** Verificar si un string cumple con una expresión regular **Método 3:** Verificar el largo de un string
		Synt. corr. deviation	
	Precision	Scale	
		Standard error	

		Granularity	
Completeness	Density	Density ratio	**Método 4:** Porcentaje de valores no nulos **Método 5:** Porcentaje ponderado de valores no nulos
	Coverage	Coverage ratio	
Freshness	Currency	Currency1	
		Currency2	
		Freshness boolean	
	Timeliness	Timeliness	
	Volatility	Volatility	
Consistency	Domain integrity	Dom. int. Rule	
	Relation integrity	Rel. int. rule	**Método 7:** Verificar si un entero es menor a un número dado y otro entero es mayor a otro número dado **Método 8:** Verificar si la edad es correcta dada la fecha de nacimiento
	Inter-Relation integrity	Inter. int. rule	
Uniqueness	Duplicate free	Non-duplicates ratio	**Método 9:** Porcentaje de no duplicados
	Contradiction free	Non-contr. ratio	

Tabla 4: Métodos implementados en el catálogo de la herramienta clasificados por la dimensión/factor/métrica que miden

5.5. Extender el proyecto.

Uno de los requisitos no funcionales son los motores de base de datos a tener en cuenta para las bases de datos que van a ser evaluadas, definimos trabajar con MySQL y PostgreSQL. Igualmente, y como vimos en el diseño de la arquitectura, buscamos un diseño para poder extender fácilmente a otros motores.

Como presentamos en el diagrama de arquitectura definimos dos módulos: *Extractor* y *Acceso a base de datos*. El módulo *Extractor* tiene la responsabilidad de conectarse a la base de datos, es decir, tiene la lógica para conectarse, iniciar y finalizar transacciones, ejecutar consultas y ejecutar sentencias. En esta clase se utiliza los jdbc DriverManager como interfaz de conexión, *com.mysql.jdbc.Driver* para MySQL y *org.postgresql.Driver* para PostgreSQL. La ventaja de utilizar esta librería de java es que expone los mismos métodos para los dos motores (MySQL y PostgreSQL), la única diferencia se da al momento de registrar el driver que implica indicar el driver del motor a utilizar. Por lo tanto, implementamos la clase *Extractor.java* que contiene todas las funciones e implementaciones de las clases *MySQL.java* y *PostgreSQL.java* que extienden la clase anterior e implementa el método de conexión a la base que es el encargado de registrar el driver. Al momento de extender la herramienta a un nuevo motor, es necesario implementar la clase que se va a encargar de extender la clase *Extractor.java* y registrar el driver correcto.

Por otro lado, tenemos el módulo *Acceso a base de datos*, este módulo tiene la lógica de consultas a la base de datos. Detectamos diferencias en las consultas SQL que hacemos dependiendo del motor, sobre todo de aquellas para obtener la información de los esquemas de las bases de datos que queremos evaluar. Por lo tanto, definimos la clase *DataAccess.java* que implementa las consultas que son comunes a los dos motores y *MySQLDataAccess.java* y *PostgreSQLDataAccess.java* la extienden para implementar los métodos que se diferencian. En el Anexo II: Funciones para extender a otro motor se detallan cual son los métodos a implementar al momento de extender la herramienta.

6. Caso de estudio

En esta sección describimos un caso de estudio. Primero se detalla la base de datos utilizada para el ejemplo, luego los pasos seguidos para armar el caso y por último los resultados obtenidos.

El sistema soporta dos tipos de métodos en los web services, el que se le pasan los datos para analizar y el que se le pasa el string de conexión a la base de datos y obtiene los datos directamente desde esta. Por lo tanto, vamos a realizar un caso de estudio que utilice los dos tipos de web service.

Para realizar el caso de estudio evaluamos diferentes bases de datos descargadas de internet. Una de ellas fue la base de datos BIRT [29] brindada por Eclipse, pero ésta no contenía errores para aplicar los conceptos de la herramienta. También analizamos la base de datos Catálogo de Bienes, Servicios y Obras [39] en la cual no detectamos errores suficientes para crear el caso de estudio. Además, en ninguno de los casos anteriores teníamos conocimiento del contexto de los datos para evaluar diferentes dimensiones, factores, métricas y métodos.

6.1. Descripción de la base de datos

Para el caso de estudio vamos a utilizar parte de una base de datos que nos proporcionaron en uno de nuestros trabajos. Esta base de datos se utilizó para hacer testing de una versión del producto que actualmente está en producción en un par de clientes. No son datos de producción y en algunos casos, sobre todo los campos de descripciones son valores dummys. Pero las relaciones entre las entidades y los valores de varios atributos son ejemplos de casos reales. Además, como es la base de datos que se utilizó para un ciclo de testing, contiene alguna inconsistencia en los datos producida por los errores de código que se encontraron en ese ciclo de pruebas.
Otra ventaja de utilizar esta base de datos para el caso de estudio es que tenemos el conocimiento pleno del dominio de la aplicación y podemos definir métricas aprovechando este conocimiento para encontrar posibles problemas de calidad que tengan los datos.

Antes de presentar la definición del modelo de calidad para el caso de estudio, vamos a realizar una breve descripción de la aplicación. Se trata de un gestor de eventos, los eventos se crean de forma manual, a partir de llamadas telefónicas o a través de mensajes. Cada evento pasa por tres etapas de atención en las que participan varios usuarios. La primera etapa es la de creación, el usuario ingresa la información disponible para tratar el evento. En la segunda etapa, que es la de gestión, el usuario accede a la información disponible y toma acciones que va registrando. Durante esta etapa pueden participar varios usuarios al mismo tiempo. En la tercera y última etapa, que es la de supervisión, se revisa toda la información ingresada en la vida del evento.

La base de datos de esta aplicación tiene alrededor de 100 tablas, por la sensibilidad de la información para el caso de estudio vamos a utilizar algunas de ellas. En la figura 12 incluimos las tablas con las que vamos a trabajar y las relaciones entre ellas.

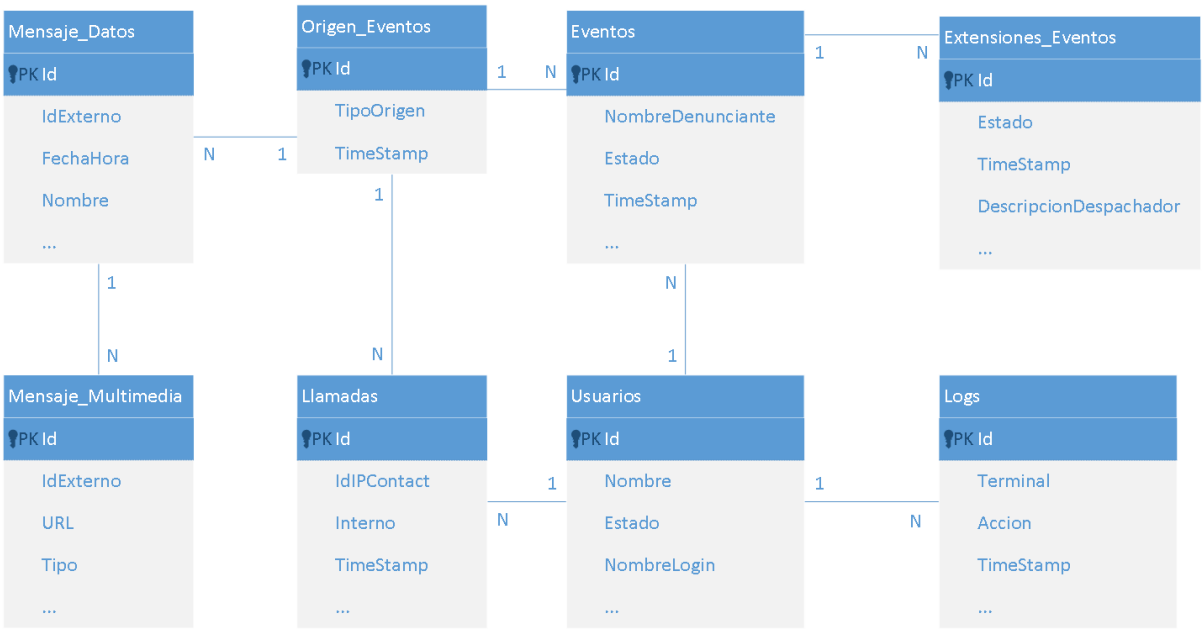


Figura 12: Modelo de dominio

Para tener una idea del tamaño de la base con la que vamos a trabajar, la tabla 5 muestra para cada tabla la cantidad de tuplas que tiene.

Tabla	Cantidad de tuplas
Eventos	1.998
Origen_Eventos	2.002
Llamadas	487
Mensaje_Datos	873
Mensaje_Multimedia	6
Extensiones_Evento	2.860
Usuarios	14
Logs	45.192

Tabla 5: Cantidad de filas de las tablas de la base de datos del caso de estudio

6.2. Definición del modelo de calidad de datos

Las definiciones de las métricas las vamos a hacer en base a algunos problemas de la base de datos y de la aplicación que conocemos. La base de datos tiene problemas de diseño que se vienen arrastrando versión tras versión. Al no tener definidas las claves foráneas y existir relaciones entre las entidades, entendemos que es un punto interesante para medir la calidad. Por otro lado, a nivel de la aplicación algunos de los cambios en la versión que estuvo a prueba fueron:

- Se elimina un estado de la tabla Extension_Eventos.
- En la tabla de Llamadas se agregó el atributo Cola.

Algunos errores de la aplicación que se vienen repitiendo en las versiones anteriores y que están pendientes de solucionar son:

- El método para obtener la IP del pc donde se está utilizando (dato importante para completar las tuplas de Logs) debe tomar la IPv4 y en algunas ocasiones toma la IPv6.
- En la tabla de Llamadas, se guarda la hora de inicio, la hora de fin y la duración de la llamada, esta información la toma de la central telefónica y a veces hay algunas diferencias (la duración no es un atributo auto calculado).

Por lo tanto, en base a los problemas que sabemos que pueden tener los datos, para el caso de estudio vamos a definir las métricas instanciadas y agregaciones que listamos en las tablas 6 y 7.

MI	**Dimensión**	**Factor**	**Métrica**	**Granularidad**	**Descripción**
1	Accuracy	Syntactic correctness	Syntactic correctness boolean	Celda	Verificar que el Estado de la tabla Extension_eventos pertenezca al conjunto {FaltaDespachar, Despachado, Cerrado, TomadoSupervisor, Archivado}
2	Completeness	Density	Density ratio	Columna	Verificar la completitud del dato Cola de la tabla Llamadas
3	Consistency	Domain integrity	Domain integrity rule	Celda	Verificar si la IP de la tabla Logs empieza con 172.31.34
4		Relation integrity	Relation integrity rule	Conjunto de celdas	Verificar que la duración de la tabla Llamadas sea igual a la resta de HoraFin y HoraInicio
5		Inter-Relation integrity	Inter-Relation integrity rule	Conjunto de columnas	Verificar la FK entre la tabla Eventos y Origen_Eventos

Tabla 6: Métricas instanciadas del modelo de calidad de datos del caso de estudio

Agregación	**MI**	**Granularidad**	**Granularidad**	

		Inicial	Final	
1	1	Celda	Columna	Promedio
2	3	Celda	Columna	Promedio
3	4	Conjunto de celdas	Conjunto de columnas	Promedio

Tabla 7: Agregaciones de modelo de calidad de datos del caso de estudio

Para comenzar con el caso de estudio lo primero que tenemos que hacer es definir el modelo de calidad, para esto tenemos que proporcionar un nombre que va a ser el que identifique al modelo de calidad y la información de la base de datos a la que le vamos a medir la calidad, en la figura 13, se muestran los campos necesarios para crear un modelo de calidad. Antes de crear el modelo se puede verificar la conexión con la base de datos a evaluar para validar la información ingresada.

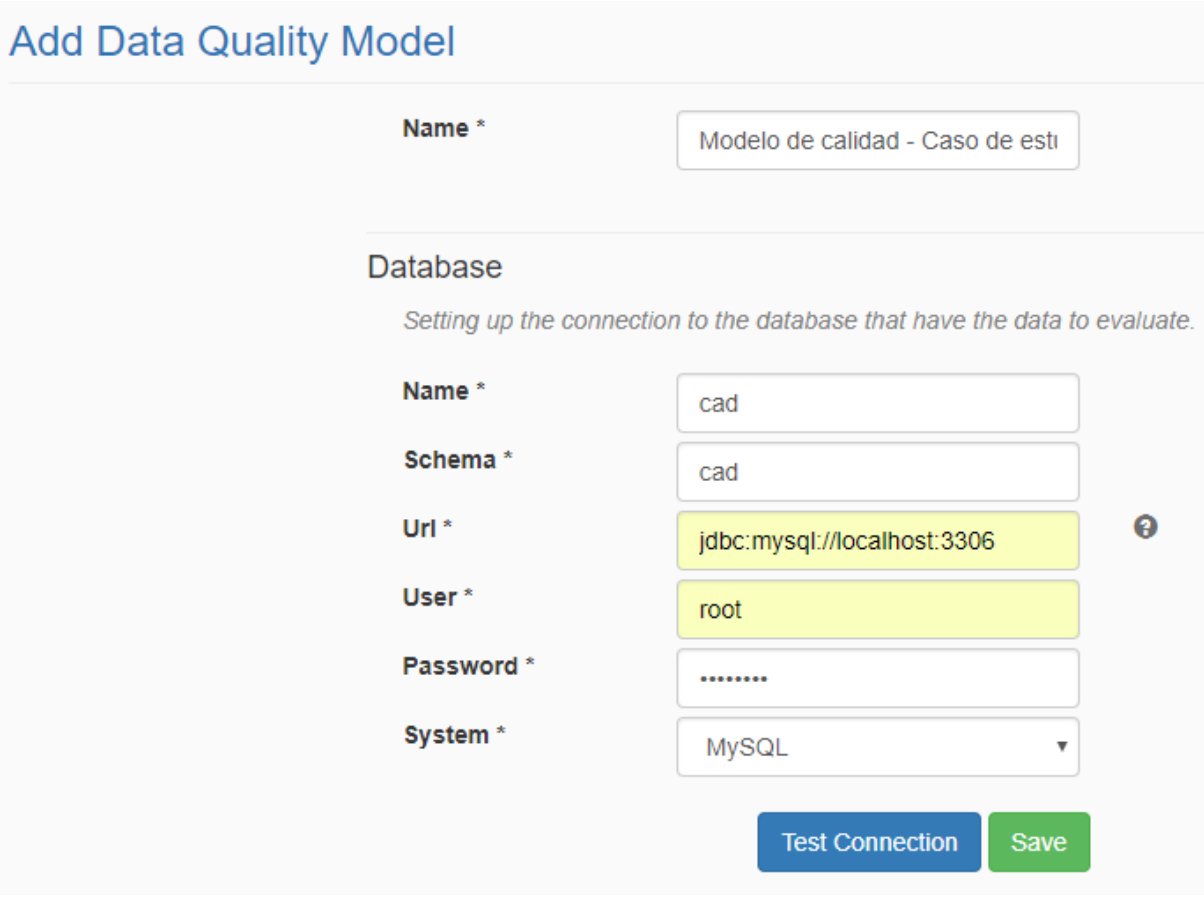


Figura 13: Creación del modelo de calidad del caso de estudio

El siguiente paso es definir las métricas instanciadas de la tabla 6. Para una de las métricas instanciadas vamos a utilizar uno de los métodos implementados en el catálogo y para el resto vamos a implementar el método. Para la primera métrica instanciada del modelo de calidad vamos a mostrar el paso a paso, el resto de las métricas instanciadas del modelo de calidad las hacemos de forma similar.

La primera métrica instanciada del modelo de calidad tiene el objetivo de verificar la correctitud sintáctica del atributo *Estado* de la tabla *Extension_Eventos*, los valores válidos son: FaltaDespachar, Despachado, Cerrado, TomadoSupervisor y Archivado. Para implementar esta métrica instanciada debemos implementar el método en el Web Service. Por lo tanto, vamos a definir la operación *isValidState* que recibe un parámetro de entrada *data* y devuelve verdadero si *data* es un estado válido, en otro caso devuelve falso. En la figura 14 mostramos la declaración de la operación.

```
@WebMethod
public boolean isValidState(
        @XmlElement(required = false, defaultValue = "null") @WebParam(name = "data") String data)
```

Figura 14: Declaración de la operación *isValidState*

Luego de implementar el método, lo tenemos que dar de alta en la herramienta para poder utilizarlo en la definición de métricas instanciadas. Para esto debemos ir al Catálogo, seleccionar la dimensión *Accuracy*, el factor *Syntactic correctness*, la métrica *Syntactic correctness boolean* y agregar el método. Para poder agregar el método tenemos que especificar el nombre, una breve descripción, la url del web service que expone las operaciones que implementan los métodos, la operación que va a implementar el método que estamos creando y por último debemos configurar los tipos de parámetros que recibe la operación. En la figura 15 mostramos cómo se da de alta el método.

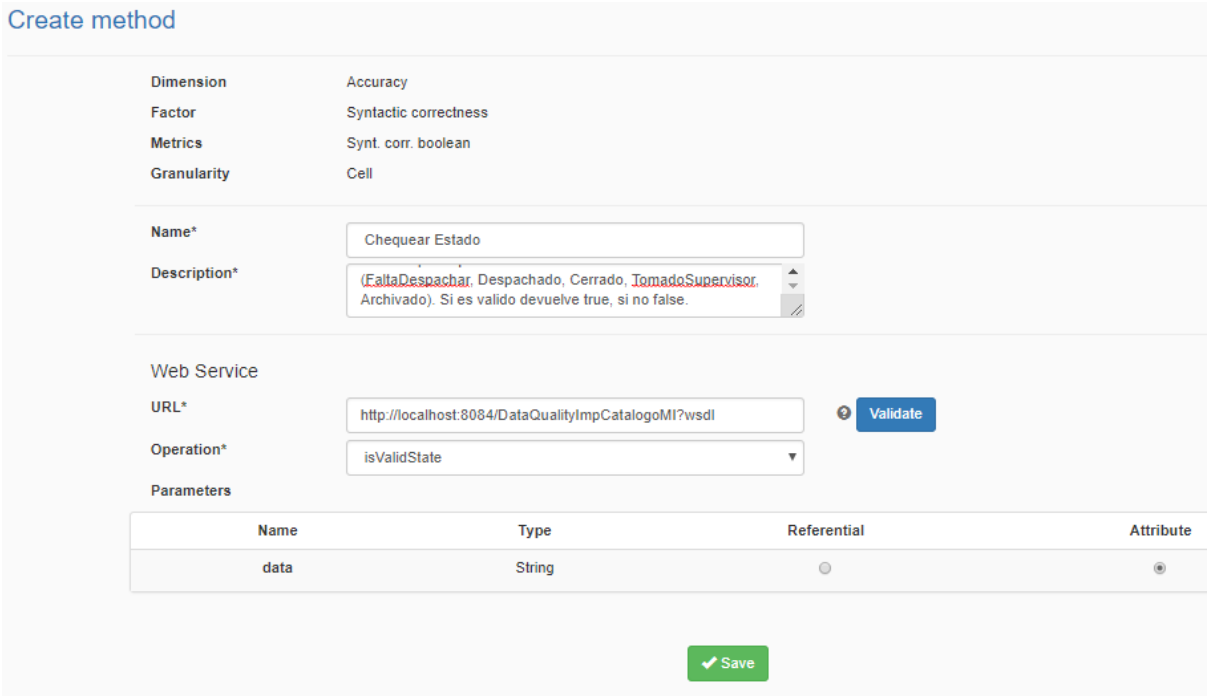


Figura 15: Creación del método Chequear Estado

Este paso tuvimos que realizarlo porque para esta métrica instanciada no utilizamos ningunos de los métodos que tenemos en el catálogo. Este nuevo método queda disponible para utilizar tanto en la métrica que vamos a definir a continuación como en cualquier otra que aplique, ya sea para esta modelo de calidad u otro. Ahora podemos continuar con la definición de la métrica instanciada, para esto debemos regresar al modelo de calidad que definimos anteriormente y crear la métrica instanciada, debemos especificar los siguientes datos: el

nombre de la métrica instanciada, la dimensión, el factor, la métrica, la granularidad y el método que la implementa. Además, debemos configurar a qué datos de la base que estamos evaluando le vamos a aplicar la métrica instanciada. En la figura 16 mostramos cómo se da de alta la métrica instanciada.

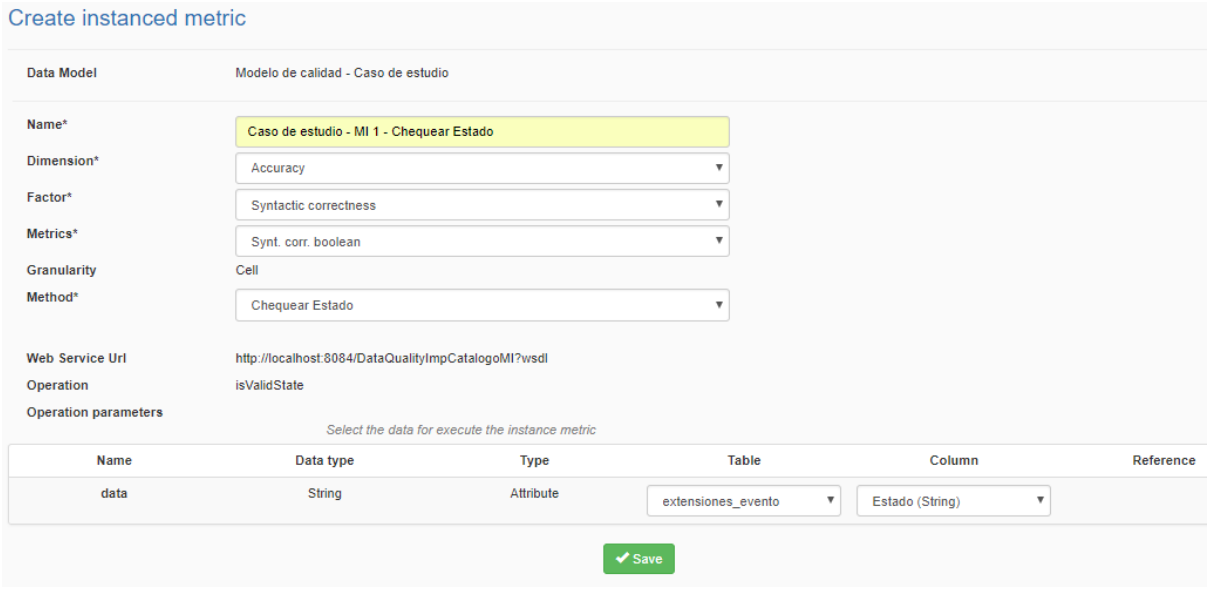


Figura16: Creación de la métrica instanciada 1 del caso de estudio

De la misma forma damos de alta las otras métricas instanciadas del modelo de calidad que definimos para el caso de estudio. En la figura 17 se muestra como se ve en la herramienta las métricas instanciadas del modelo de calidad.

Figura 17: Métricas instanciadas del modelo de calidad del caso de estudio

Para continuar con la definición del modelo de calidad del caso de estudio tenemos que dar de alta las agregaciones. A continuación, vamos a mostrar el paso a paso de la primera agregación. Para dar de alta una agregación debemos especificar el nombre de la agregación, sobre los resultados de qué métrica instanciada se aplica, la granularidad final, la URL del web service que implementa la agregación y la operación que se utilizará. Una vez seleccionada la operación hay que definir qué tipo de parámetros es cada uno de los parámetros de la operación. En la figura 18 se muestra la creación de la primera agregación.

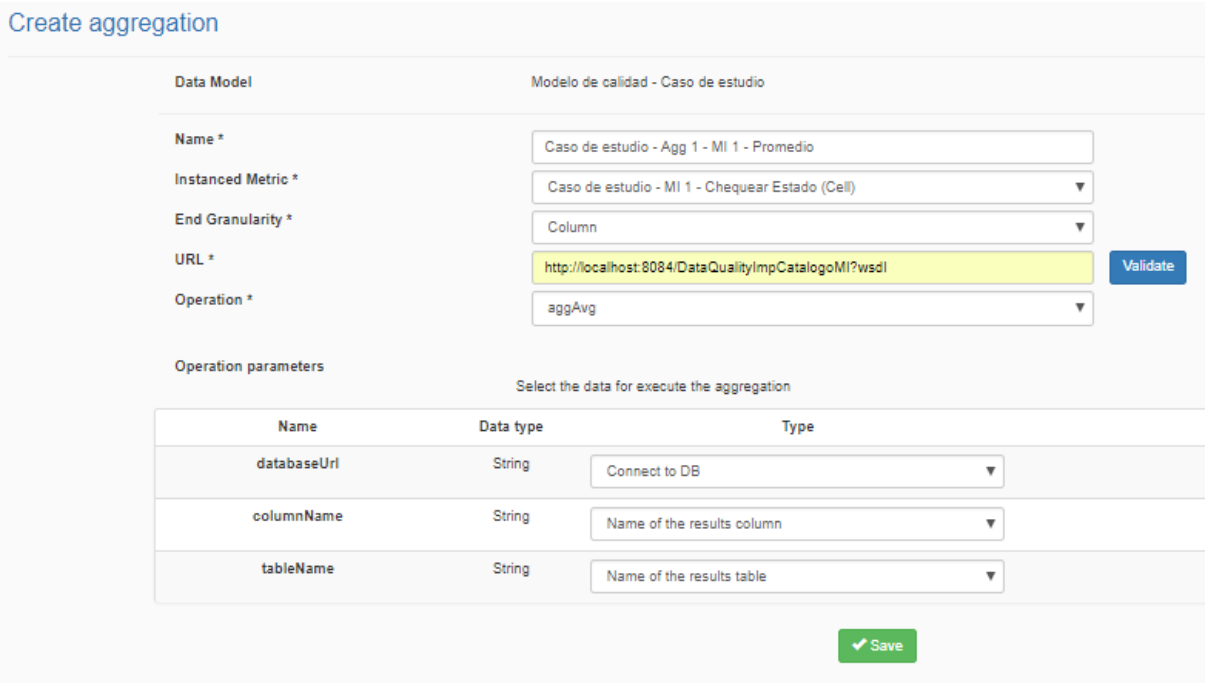


Figura 18: Creación de la agregación 1 del modelo de calidad del caso de estudio

De la misma forma que la anterior definimos el resto de las agregaciones. En la figura 19 se muestra como quedan definidas en la herramienta la definición de todas las agregaciones del modelo de calidad.

Figura 19: Agregaciones del modelo de calidad del caso de estudio

Finalmente, con la definición de las métricas instanciadas y las agregaciones completamos el modelo de calidad de datos para el caso de estudio. Desde la herramienta es posible generar un pdf con la definición del modelo, lo incluimos en el Anexo III: Definición del modelo de calidad del caso de estudio.

6.3. Ejecución del modelo de calidad de datos

Para continuar con el caso de estudio primero tenemos que ejecutar las métricas instanciadas, para esto tenemos que ir a la opción del menú de ejecutar métricas instanciadas y seleccionar el modelo de calidad de datos con el que vamos a trabajar. En la figura 20 se listan las métricas instanciadas del modelo de calidad, debemos seleccionar cuales vamos a ejecutar.

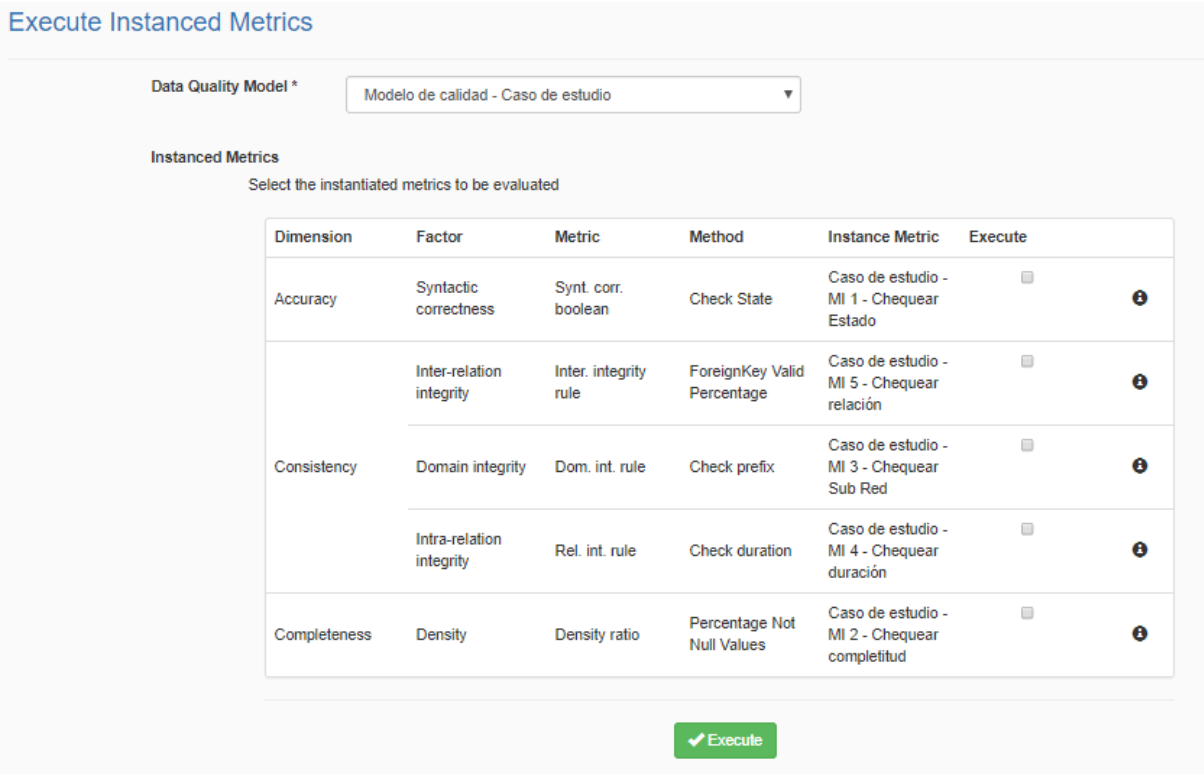


Figura 20: Selección de métricas instanciadas del modelo de calidad del caso de estudio para ejecutar

Para el caso de estudio vamos a ejecutar todas las métricas instanciadas que definimos en el modelo de calidad. En las figuras 21, 22, 23, 24 y 25 vemos los resultados de la ejecución de las métricas instanciadas del modelo. Además, en el Anexo IV: Reporte de ejecución de las métricas instanciadas del modelo de calidad datos del caso de estudio incluimos el reporte de ejecución, para las métricas de granularidad celda, conjunto de celda o tupla incluimos solo la primera página.

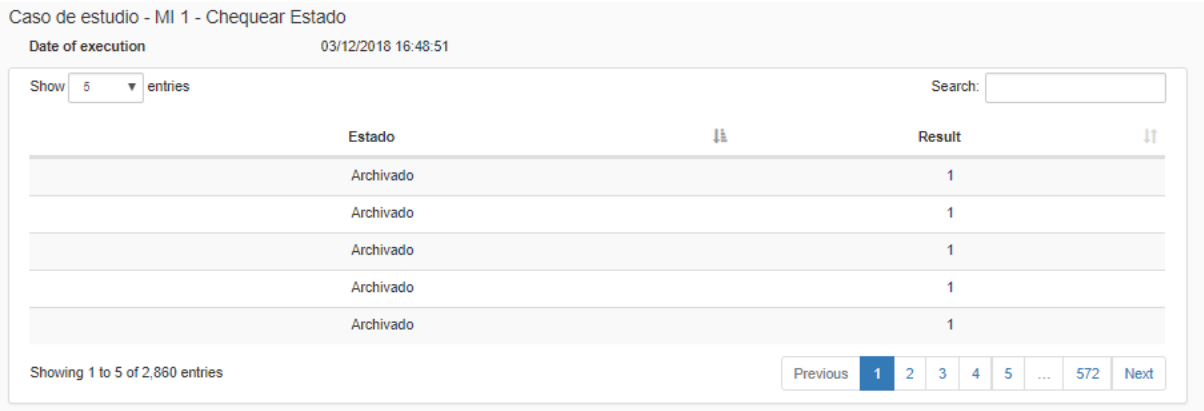
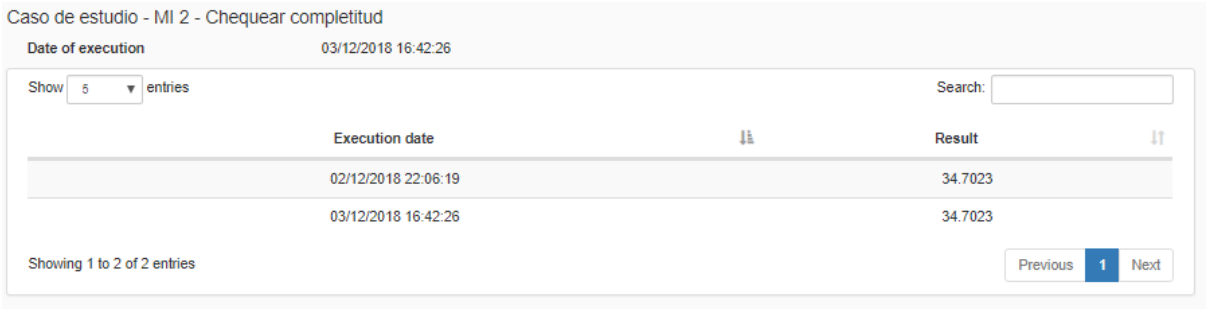


Figura 21: Resultado de la ejecución de la métrica instanciada 1


Caso de estudio - MI 2 - Chequear completitud

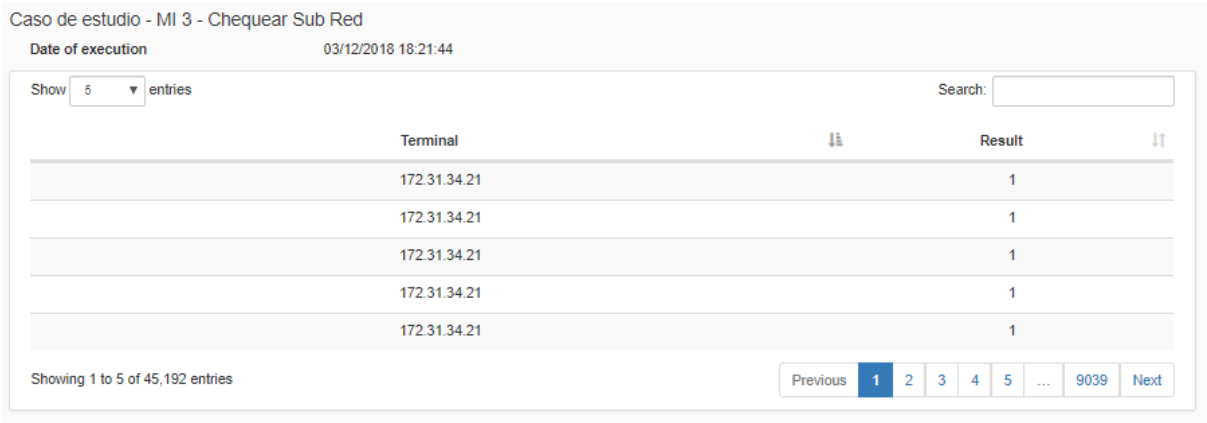
Date of execution 03/12/2018 16:42:26

Show 5 entries Search:

Execution date	Result
02/12/2018 22:06:19	34.7023
03/12/2018 16:42:26	34.7023

Showing 1 to 2 of 2 entries Previous 1 Next

Figura 22: Resultado de la ejecución de la métrica instanciada 2


Caso de estudio - MI 3 - Chequear Sub Red

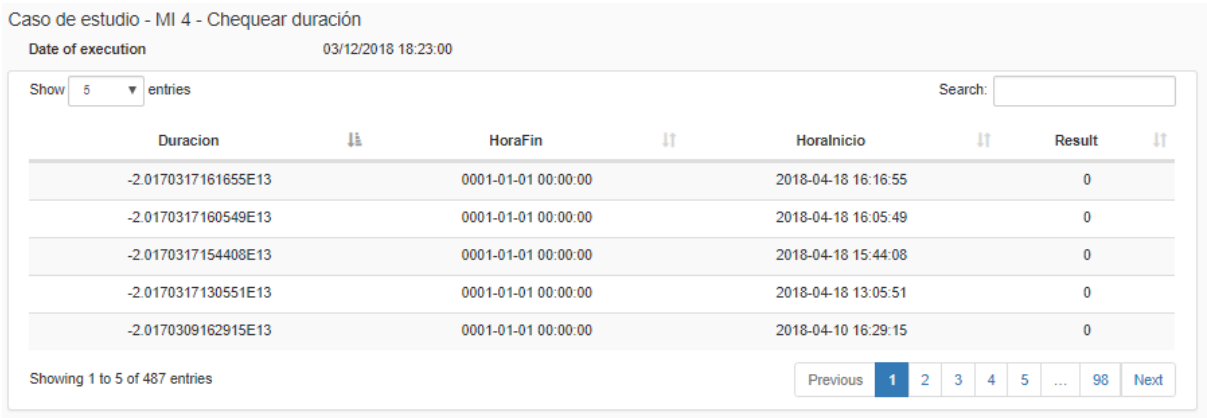
Date of execution 03/12/2018 18:21:44

Show 5 entries Search:

Terminal	Result
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1

Showing 1 to 5 of 45,192 entries Previous 1 2 3 4 5 ... 9039 Next

Figura 23: Resultado de la ejecución de la métrica instanciada 3


Caso de estudio - MI 4 - Chequear duración

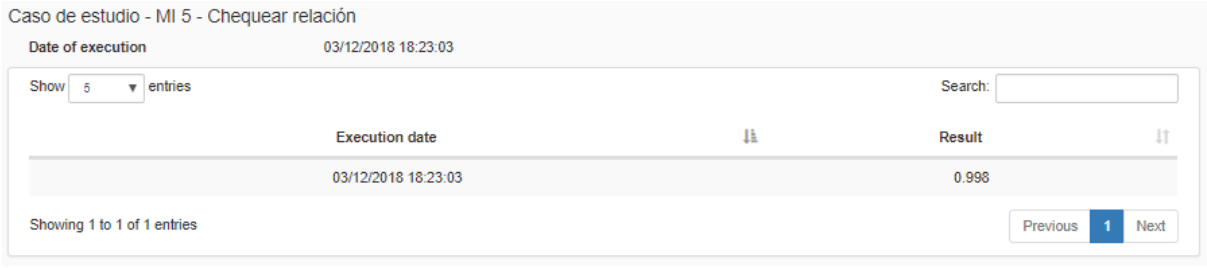
Date of execution 03/12/2018 18:23:00

Show 5 entries Search:

Duracion	HoraFin	HoraInicio	Result
-2.0170317161655E13	0001-01-01 00:00:00	2018-04-18 16:16:55	0
-2.0170317160549E13	0001-01-01 00:00:00	2018-04-18 16:05:49	0
-2.0170317154408E13	0001-01-01 00:00:00	2018-04-18 15:44:08	0
-2.0170317130551E13	0001-01-01 00:00:00	2018-04-18 13:05:51	0
-2.0170309162915E13	0001-01-01 00:00:00	2018-04-10 16:29:15	0

Showing 1 to 5 of 487 entries Previous 1 2 3 4 5 ... 98 Next

Figura 24: Resultado de la ejecución de la métrica instanciada 4


Caso de estudio - MI 5 - Chequear relación

Date of execution 03/12/2018 18:23:03

Show 5 entries Search:

Execution date	Result
03/12/2018 18:23:03	0.998

Showing 1 to 1 of 1 entries Previous 1 Next

Figura 25: Resultado de la ejecución de la métrica instanciada 5

Para los casos de métricas instanciadas de granularidad celda, conjunto de celdas o tupla, cada vez que se ejecutan se borran los resultados de las ejecuciones anteriores, en el caso de estudio son las métricas 1, 3 y 4. Para el resto de las granularidades se guardan el

resultado de todas las ejecuciones junto con la fecha de ejecución. En el caso de estudio la métrica instanciada 2 tenía una ejecución previa mientras que la métrica instanciada 5 tiene una sola.

Una vez que ejecutamos las métricas instanciadas, es momento de ejecutar las agregaciones que habíamos definido, para esto vamos a la opción del menú para ejecutar las agregaciones. Tal como sucede con la ejecución de métricas instanciadas, se listan las agregaciones definidas en el modelo de calidad de datos para que el usuario seleccione cual desea ejecutar (figura 26).

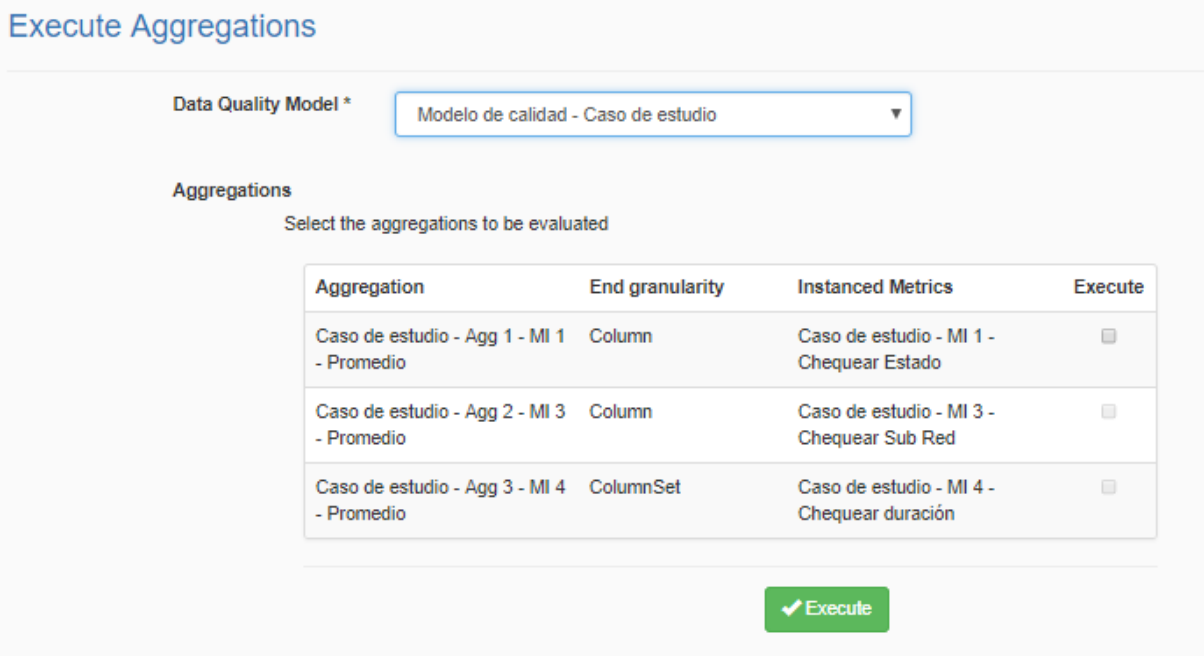


Figura 26: Lista de agregaciones para ejecutar

Una vez ejecutadas las agregaciones se muestra en pantalla el resultado (figuras 27, 28, 29) y se puede descargar el reporte, el mismo lo incluimos en el [Anexo V: Reporte de ejecución de las agregaciones del modelo de calidad datos del caso de estudio.]

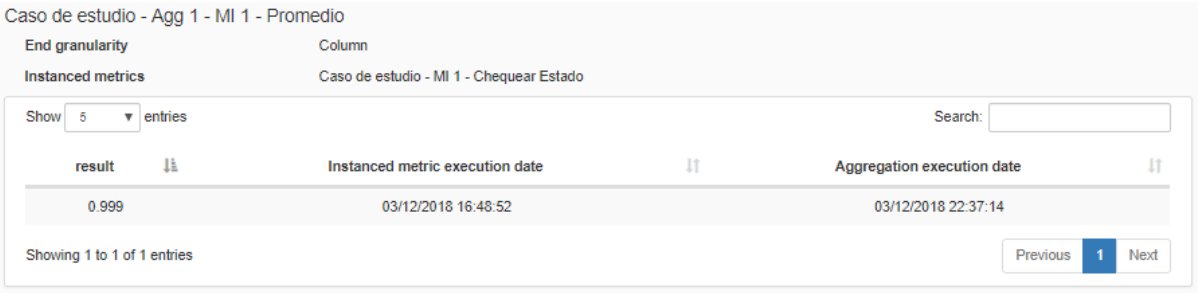


Figura 27: Resultado de la ejecución de la agregación 1 del modelo de calidad de datos del caso de estudio

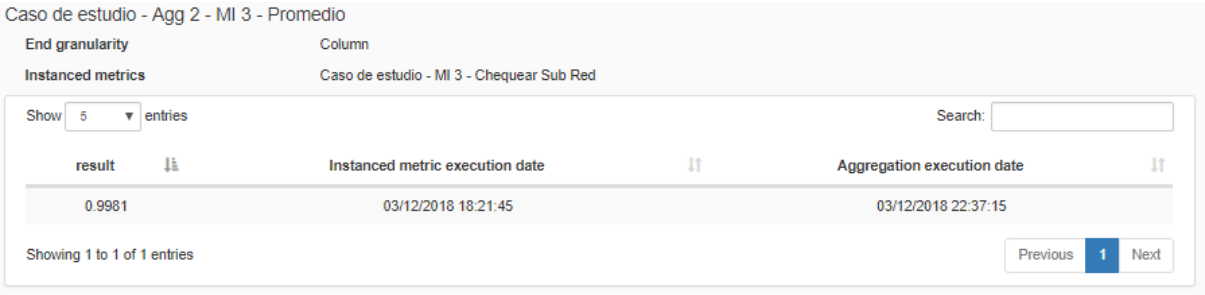


Figura 28: Resultado de la ejecución de la agregación 2 del modelo de calidad de datos del caso de estudio

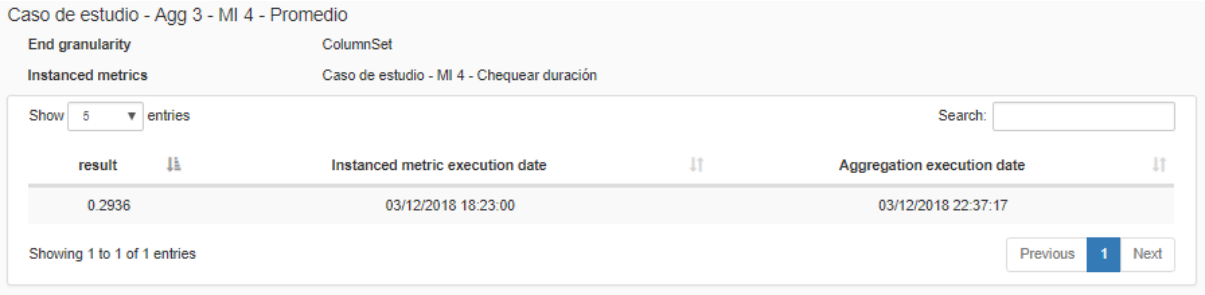


Figura 29: Resultado de la ejecución de la agregación 3 del modelo de calidad de datos del caso de estudio

Al momento de ejecutar las agregaciones nos da la opción de elegir si queremos conservar los datos de ejecución de las métricas instanciadas o no, en este caso no los conservamos.

Hasta ahora no habíamos incluido en el caso de estudio la definición de métricas instanciadas compuestas. Lo que buscamos con esta métrica instanciada compuesta es reutilizar los resultados de la métrica instanciada que ejecutamos previamente o reutilizar métodos implementados y realizar chequeos extras. Para el caso de estudio vamos a tomar la tabla llamadas, previamente habíamos definido una métrica instanciada (Caso de estudio - MI 4 - Chequear duración) que toma la fecha de inicio de la llamada y la fecha de fin y chequea si la diferencia es igual a la duración. Definimos una nueva métrica instanciada que verifique que la duración sea mayor o igual a cero. Luego para obtener un único valor de calidad para la duración podemos combinar esas dos métricas instanciadas y obtener un único valor de calidad para el atributo duración de la tabla llamadas. En la figura 30 se muestra cómo se crea en la herramienta una métrica instanciada compleja. Debemos especificar el nombre, las métricas instanciadas que van a participar (dos o más), la granularidad final y el WS que la va a implementar. Una vez seleccionada la operación del WS debemos especificar qué tipo de parámetros es cada uno, las opciones son: resultado de la ejecución de la métrica instanciada (Result of) o referencia (Reference). Queda por fuera del alcance del proyecto la ejecución.

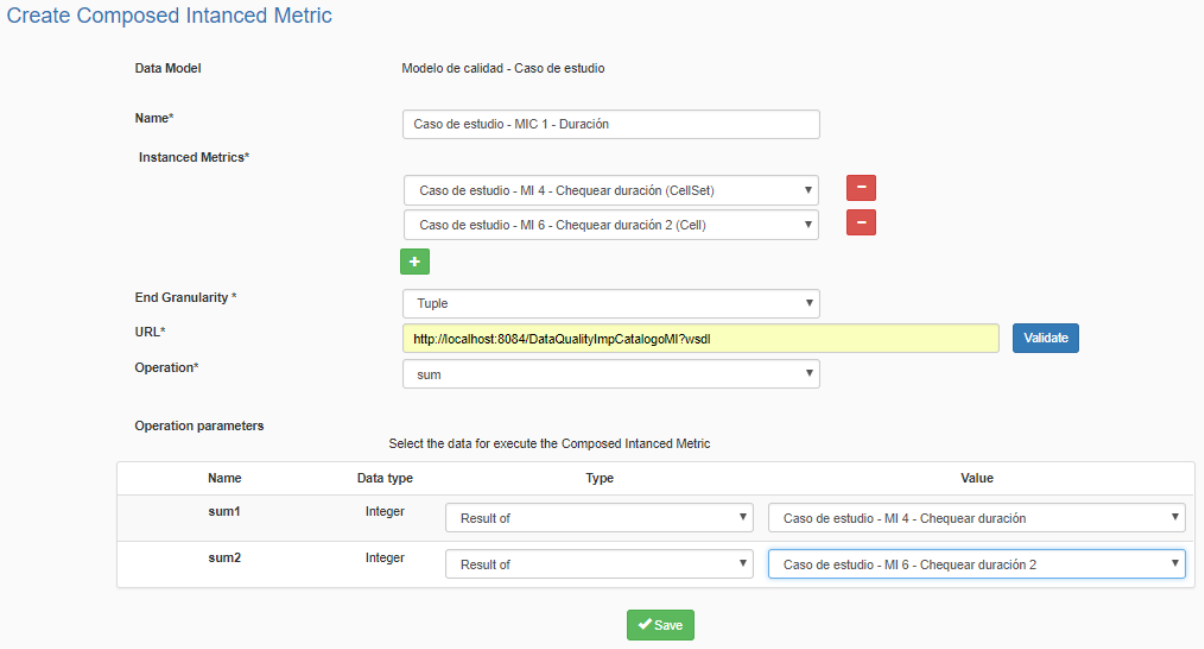


Figura 30: Creación de una métrica instanciada compuesta

Por último, mencionar que se puede acceder directamente a los resultados de las ejecuciones tanto de las métricas instanciadas como de las agregaciones desde la sección del menú resultados.

7. Conclusiones y trabajo a futuro

En esta sección presentamos las conclusiones obtenidas a partir del trabajo realizado, las mejoras que se pueden implementar a la herramienta desarrollada y algunas líneas de trabajo que detectamos y nos parece interesante para que se puedan trabajar a futuro.

7.1. Conclusiones

Se desarrolló una herramienta para la gestión de calidad de datos que permite definir un catálogo de métricas de calidad, modelos de calidad de datos y ejecutar los modelos de calidad para obtener una evaluación de la calidad de los datos. Se brinda un catálogo con la especificación de algunas métricas y la implementación de algunos métodos, la implementación se realizó de forma genérica por lo cual pueden ser reutilizadas para evaluar diferentes conjuntos de datos. Además, la implementación de los métodos se realizó en web services externos a la herramienta, brindando de esta forma la flexibilidad en la implementación de los métodos de medición sin quedar limitados a un lenguaje especifico de la herramienta o a los métodos del catálogo.
La herramienta tiene la flexibilidad para evaluar bases de datos de diferentes motores, en el proyecto se incluyó la implementación para MySQL y PostgresSQL. En la documentación se detalla la implementación necesaria en Java de la interfaz de acceso al motor de base de datos para extenderlo a otros motores. Cabe destacar, que dicha implementación es mínima y en general solo es necesario adaptar el código SQL que se invoca en los métodos de la interfaz.
Mediante las agregaciones es posible generar un resumen de los resultados de las evaluaciones de manera que se pueda tener un historial para visualizar la evolución de la calidad de datos. Las métricas instanciadas compuestas permiten combinar los resultados de diferentes métricas de forma que se puede obtener un único resultado de diferentes evaluaciones para un mismo conjunto de datos.
Aunque no era un requerimiento de proyecto, nos preocupaba el volumen de los datos que se podría llegar a evaluar con la herramienta, en el caso de estudio se evaluó un conjunto de datos de 45.192 registros y la herramienta funcionó correctamente llegando a retornar el resultado de la evaluación.
La configuración inicial para poder ser utilizada es mínima, solo es necesario configurar el servidor de base de datos donde se almacenan los datos de la herramienta y los metadatos.

Uno de los principales desafíos que nos encontramos en el diseño de la aplicación fue como identificar los registros en caso que no tengan claves primarias. Resolvimos realizar una copia de los datos a evaluar y crear para cada registro una clave, de esta forma se puede asociar el resultado de la evaluación a los datos evaluados. A su vez, esto nos permite tener los datos de la última evaluación, de manera que si en la base de datos original los datos son modificados igualmente podemos ver cuales los datos que fueron evaluados.

7.1.1. Limitaciones

Solo se almacenan los datos utilizados para la última evaluación, lo cual no permite ver el historial completo de todas las evaluaciones, pero como ya se mencionó, se puede crear un resumen de las evaluaciones.
Las pruebas de la herramienta se realizaron con los métodos del catálogo y del caso de estudio, no se realizaron pruebas excautivas con web services disponibles en internet. Existen diversas especificaciones de web services SOAP para evaluar la calidad de datos, en la implementación del proyecto no se cubrieron todos los casos, aunque se intentó abarcar la mayor cantidad posible. En la implementación no se incluye la autenticación y configuraciones de seguridad que tienen algunos de los webs services disponibles en internet.
Al realizar una copia de los datos a evaluar, si la base de datos es grande, los tiempos de copiado pueden ser grandes.

7.2. Trabajo a futuro

Durante el desarrollo del proyecto surgieron diferentes propuestas que debido al alcance del proyecto quedaron por fuera y se podrían considerar como mejoras o trabajo a futuro.

La implementación de la ejecución de métricas instanciadas compuesta y la administración de usuarios quedó por fuera del alcance del proyecto. También se había planteado que al ejecutar nuevamente una métrica instanciada compuesta los resultados de la evaluación anterior se resuman automáticamente, esto no se llegó a implementar.
Los resultados de las ejecuciones se pueden visualizar en forma de tablas en la herramienta y descargar un .pdf, entendemos que sería una mejora poder ver esos resultados y las comparaciones con los históricos en distintos formatos de graficas o dashboard para facilitar la lectura de los usuarios de la herramienta.

La herramienta se podría extender para que permita definir varios catálogos, así como también generar un nuevo módulo para que la implementación de los métodos se realice en la misma herramienta. Se podría considerar realizar el proyecto para base de datos no relacionales e incluir el consumo de web services RESTfull.
En base a las herramientas que evaluamos antes de comenzar el proyecto, una funcionalidad que vimos en varias de ellas era el perfilado de los datos, esto puede ser de ayuda para comenzar la definición de un modelo de calidad. Al conocer la base de datos del modelo de calidad de datos que el usuario quiere definir, se puede realizar una evaluación general para ayudar al usuario a detectar los posibles problemas que presentan sus datos. También puede ser interesante incluir alguna funcionalidad para corregir los datos desde la herramienta.

Bibliografía

[1] Scannapieco, M., y Catarci, T. (2002). Data quality under a computer science perspective. *Journal of The ACM – JACM, 2*.

[2] Batini, C., y Scannapieca, M. (2006). Data Quality: Concepts, Methodologies and Techniques. Berlin: Springer.

[3] Eckerson, W. (2002). Data Quality and the Bottom Line. *TDWI Report Series*.

[4] Wang, R. Y., y Strong, D. M. (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems, 12(4)*, 5.

[5] Curso: Calidad de Datos - Facultad de Ingeniería de la UdelaR. https://www.fing.edu.uy/inco/cursos/caldatos/index.php, accedida abril 2016.

[6] Pipino, L. L., Lee, Y. W., y Wang, R. Y. (2002). Data Quality Assessment. *Communications of the ACM, 45(4)*, 211–218.

[7] Wang, R. Y., Reddy, M. P., y Kon, H. B. (1995). Toward quality data: An attribute-based approach. *Decision Support Systems, 13*.

[8] Basili, V. R., Caldeira, G., y Rombach, H. D. (1994). Goal question metric paradigm. *Encyclopedia of software engineering, 1*, 528-532.

[9] Naumann, F. (2013). Data profiling revisited. *SIGMOD Record*, *42*, 40-49.

[10] Koziolek, H. (2008). Goal, Question, Metric. *Dependability Metrics, 39*.

[11] Lee, Y. W., Strong, D. M., Kahn, B. K., y Wang, R. Y. (2002). AIMQ: a methodology for information quality assessment. *Information & Management, 40*, 133–146.

[12] González, L., Peralta, V., Bouzeghoub, M., y Ruggia, R. (2009). Qbox-Services: Towards a Service-Oriented Quality Platform. *Advances in Conceptual Modeling - Challenging Perspectives*, 232.

[13] Talend – Open Source Solutions. https://www.talend.com/products/talend-open-studio/#t2, accedida agosto 2016.

[14] Data Quality Services. https://msdn.microsoft.com/en-us/library/ff877917.aspx, accedida agosto 2016.

[15] Master Data Services and Data Quality Services Features Supported by the Editions of SQL Server 2016. https://msdn.microsoft.com/en-us/library/mt490149.aspx, accedida agosto 2016.

[16] DataCleaner 5.1. https://datacleaner.org/, accedida agosto 2016.

[17] GitHub - DataCleaner. https://github.com/datacleaner/DataCleaner, accedida agosto 2016.

[18] Abedjan, Z., Chu, X., Deng, D., Fernandez, R. C., Ilyas, I. F., Ouzzani, M., Papotti, P., Stonebraker, M., y Tang, N. (2016). Detecting data errors: where are we and what needs to be done?. *Proc. VLDB Endow*. 9, 12, 993-1004.

[19] Eclipse IDE for Java EE Developers. http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/neonr, accedida setiembre 2016.

[20] Apache Maven. https://maven.apache.org, accedida setiembre 2016.

[21] Web MVC framework. https://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html, accedida octubre 2016.

[22] Bootstrap. http://getbootstrap.com, accedida octubre 2016.

[23] MySQL. https://www.mysql.com, accedida octubre 2016.

[24] Git. https://git-scm.com, accedida noviembre 2016.

[25] Bitbucket. https://bitbucket.org, accedida noviembre 2016.

[26] Hibernate. http://hibernate.org, accedida octubre 2016.
[27] iText. http://itextpdf.com, accedida febrero 2017.
[28] Predic8. https://www.predic8.com, accedida febrero 2017.
[29] BIRT – Sample Database. http://www.eclipse.org/birt/documentation/sample-database.php, accedida julio 2017.
[30] Generatedata.com. http://www.generatedata.com, accedida julio 2017.
[31] MySQL – Connectors. https://dev.mysql.com/doc/connectors/en/connector-j-reference-type-conversions.html, accedida agosto 2017.
[32] MySQL – Data Types. https://dev.mysql.com/doc/refman/5.7/en/data-types.html, accedida agosto 2017.
[33] PostgreSQL – Data Types. https://www.postgresql.org/docs/9.2/static/datatype.html, accedida agosto 2017.
[34] Apache Tomcat 7. https://tomcat.apache.org, accedida noviembre 2018.
[35] Apache Tomcat 7 - Tomcat Setup. https://tomcat.apache.org/tomcat-7.0-doc/setup.html, accedida noviembre 2018.
[36] Apache Tomcat 7 - Tomcat Web Application Deployment. https://tomcat.apache.org/tomcat-7.0-doc/deployer-howto.html, accedida noviembre 2018.
[37] Java SE Development Kit 8 Downloads. https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html, accedida noviembre 2018.
[38] Walber. (2014). Precision and recall [Figura]. https://commons.wikimedia.org/wiki/File:Precisionrecall.svg, accedida diciembre 2018.
[39] ACCE - Catálogo de Bienes, Servicios y Obras https://www.comprasestatales.gub.uy/inicio/datos-abiertos/catalogo-bienes-servicios-obras, accedida octubre 2017.

Anexo I: Casos de Uso

En este anexo especificamos los casos de uso de las principales funcionalidades de la herramienta.

Crear método

Objetivo: Permitir dar de alta un método en el sistema.
Actores: Usuario del sistema.
Precondiciones: Existen dimensiones, factores y métricas en el sistema.
Descripción: El caso de uso comienza cuando el usuario indica al sistema crear un método. El usuario indica el nombre, la URL del web service, la operación y los tipos de los parámetros de la operación.
Flujo normal:

1. El usuario selecciona crear método
2. El sistema despliega las dimensiones.
3. El usuario selecciona una dimensión.
4. El sistema despliega los factores de la dimensión seleccionada.
5. El usuario selecciona un factor.
6. El sistema despliega las métricas del factor seleccionado.
7. El usuario selecciona una métrica.
8. El usuario ingresa el nombre del método y la URL del web service.
9. El sistema valida el nombre del método, la URL del web service y despliega sus operaciones.
10. El usuario selecciona una operación.
11. El sistema despliega los parámetros de la operación y las opciones para seleccionar el tipo de cada parámetro.
12. El usuario selecciona el tipo de cada parámetro.
13. El sistema valida los tipos de los parámetros seleccionados.
14. El sistema crea el método.
15. Fin de CU.

Flujo alternativo:

9.a El nombre es vacío o existe en el sistema otro método con el mismo nombre.

9.a.1 El sistema despliega un mensaje de error.
9.a.2 Sigue en el punto 8 del flujo principal.

9.b. El usuario no ingresa la URL del web service o no es válida.

9.b.1. El sistema despliega un mensaje de error indicando que debe ingresar obligatoriamente la URL del web service o que no es válida.
9.b.2. Sigue en el punto 8 del flujo principal.

11.a. La métrica es de granularidad Celda, Conjunto de celda o Tupla.

11.a.1. El sistema despliega las opciones referencial y atributo para seleccionar el tipo de cada parámetro.
11.a.2. Sigue en el paso 12.

11.b. La métrica es de granularidad Columna o Conjunto de columna.

11.b.1. El sistema despliega las opciones referencial, nombre de columna, nombre de tabla y parámetro de conexión a la base de datos para seleccionar el tipo de cada parámetro.

11.b.2. Sigue en el paso 12.

11.c. La métrica es de granularidad Tabla.

11.c.1. El sistema despliega las opciones referencial, nombre de tabla y parámetro de conexión a la base de datos para seleccionar el tipo de cada parámetro.

11.c.2. Sigue en el paso 12.

11.d. La métrica es de granularidad Base de datos.

11.d.1. El sistema despliega las opciones referencial y parámetro de conexión a la base de datos para seleccionar el tipo de cada parámetro.

11.d.2. Sigue en el paso 12.

13.a. La métrica es de granularidad Columna o Conjunto de columna y el usuario no selecciona el parámetro de conexión a la base de datos, al menos un parámetro del nombre de columna y al menos un parámetro de nombre de tabla.

13.a.1. El sistema despliega un mensaje de error indicando que debe ingresar un parámetro de conexión a la base de datos, los parámetros del nombre de columna y los parámetros del nombre de tabla.

13.a.2. Sigue en el paso 12.

13.b. La métrica es de granularidad Tabla y el usuario no selecciona el parámetro de conexión a la base de datos y un parámetro del nombre de la tabla.

13.b.1. El sistema despliega un mensaje de error indicando que debe seleccionar el parámetro de conexión a la base datos y el del nombre de la tabla.

13.b.2. Sigue en el paso 12.

13.c. La métrica es de granularidad Base de datos y el usuario no selecciona el parámetro de conexión a la base de datos.

13.c.1. El sistema despliega un mensaje de error indicando que debe seleccionar el parámetro de conexión a la base de datos.

13.c.2. Sigue en el paso 12.

13.d. El usuario no selecciona los tipos para todos los parámetros de la operación.

13.d.1. El sistema despliega un mensaje de error indicando que debe seleccionar los tipos para todos los parámetros.

13.d.2. Sigue en el paso 12.

15.a. El usuario no ingresa algunos de los campos obligatorios del sistema.

15.a.1. El sistema despliega un mensaje de error indicando que debe ingresar obligatoriamente el nombre del método.

15.a.2. Sigue en el punto 8 del flujo principal.

G1. El usuario cancela la operación.

G1.1. Fin del CU.

Crear métrica instanciada

Objetivo: Crear una métrica instanciada en el sistema asociada a un modelo de calidad.
Actores: Usuario del sistema.
Precondiciones: Existe el modelo de calidad y métodos.
Descripción: El caso de uso comienza cuando el usuario indica al sistema crear una métrica instanciada.
El usuario indica el nombre, el método y los datos de cada parámetro del método.
Flujo normal:

1. El usuario selecciona Crear métrica instanciada.
2. El sistema despliega los modelos de datos.
3. El usuario selecciona el modelo de datos.
4. El usuario ingresa el nombre de la métrica instanciada.
5. El sistema valida el nombre y despliega las dimensiones.
6. El usuario selecciona una dimensión.
7. El sistema despliega los factores de la dimensión seleccionada.
8. El usuario selecciona un factor.
9. El sistema despliega las métricas del factor seleccionado.
10. El usuario selecciona una métrica.
11. El sistema despliega los métodos de la métrica.
12. El usuario selecciona el método.
13. El sistema despliega los parámetros de la operación del método.
14. El usuario ingresa los valores para los parámetros de la operación.
15. El sistema valida la configuración ingresada por el usuario y crea la métrica instanciada.
16. Fin del CU.

Flujo alternativo:

5.a. El nombre es vacío o existe en el sistema otra métrica instanciada con el mismo nombre.

5.a.1. El sistema muestra un mensaje de error.
5.b.2. Vuelve al paso 4 del flujo principal.

14.a. El parámetro es del tipo Atributo o Nombre de la columna.

14.a.1. El sistema lista el nombre de las tablas.
14.a.2. El usuario selecciona la tabla.
14.a.3. El sistema lista el nombre de las columnas.
14.a.4. El usuario selecciona el nombre de la columna.
14.a.5. Sigue en el punto 14 del flujo principal.

14.b. El parámetro es del tipo Nombre de la tabla.

14.b.1. El sistema lista el nombre de las tablas.
14.b.2. El usuario selecciona la tabla.
14.b.3. Sigue en el punto 14 del flujo principal.

14.c. El parámetro es del tipo Referencial.

14.c.1. El usuario ingresa un valor del mismo tipo de datos que el parámetro.

14.c.2. Sigue en el punto 14 del flujo principal.

14.d. El parámetro es del tipo nombre de la tabla y el usuario no selecciona la tabla.

14.d.1. El sistema muestra un mensaje de error indicando que debe seleccionar el nombre.

14.d.2. Sigue en el punto 14 del flujo principal**.**

14.e. No hay más parámetros para ingresar su valor

14.e.1. Sigue en el punto 15 del flujo principal.

15.a. La granularidad del método es Tupla. El usuario no selecciona la misma tabla para todos los parámetros, selecciona una columna para más de un parámetro o no selecciona todas las columnas de la tabla.

15.a.1. El sistema muestra un mensaje indicando el error.

15.a.2. Sigue en el punto 14 del flujo principal.

15.b. La granularidad del método es Conjunto de celdas. El usuario no selecciona la misma tabla para todos los parámetros.

15.a.1. El sistema muestra un mensaje indicando el error.

15.a.2. Sigue en el punto 14 del flujo principal.

G1. El usuario cancela la operación.

G1.1. Fin del CU.

Ejecutar métricas instanciadas y agregaciones

Objetivo: Ejecutar las métricas instanciadas y agregaciones seleccionadas por el usuario que están definidas en un modelo de calidad.
Actores: Usuario del sistema.
Precondiciones: Existe el modelo de calidad.
Descripción: El caso de uso comienza cuando el usuario indica al sistema que quiere ejecutar un modelo de calidad previamente definido.
El usuario puede seleccionar qué métricas instanciadas o agregaciones quiere ejecutar.
Flujo normal:

1. El usuario indica si quiere ejecutar métricas instanciadas o agregaciones.
2. El usuario selecciona el modelo de calidad.
3. El sistema despliega las métricas instanciadas o las agregaciones que están definidas para ese modelo de calidad.
4. El usuario selecciona las métricas instanciadas o las agregaciones que quiere ejecutar.
5. El sistema realiza una copia de los datos que va a evaluar a la base de metadatos.
6. El sistema ejecuta las métricas o agregaciones seleccionadas por el usuario y guarda los resultados en la base de metadatos.
7. El sistema le muestra al usuario los resultados de la ejecución.
8. El usuario indica que quiere exportar los resultados de la ejecución.
9. El sistema descarga un archivo con los resultados.
10. El usuario indica que quiere imprimir los resultados de la ejecución.

11. El sistema imprime los resultados de la ejecución.
12. Fin del CU.

Flujo alternativo:

5.a. Las agregaciones no tienen el resultado previo necesario para ejecutarse.

5.a.1. El sistema le informa al usuario que las métricas instanciadas asociadas a las agregaciones no fueron ejecutadas previamente y que no puede ejecutarlas.

5.a.2. Sigue en el punto 11 del flujo principal.

7.a. El usuario no quiere exportar los resultados.

7.a.1. Sigue en el punto 9 del flujo principal.

9.a. El usuario no quiere imprimir los resultados.

9.a.1. Sigue en el punto 11 del flujo principal.

G1. El usuario cancela la operación.

G1.1. Fin del CU.

Anexo II: Extender a otro motor

A continuación, se listan los métodos que hay que implementar para extender la herramienta para otro motor y se brinda una breve descripción de estos.

```
public List<GenericObject> getPKColumnsAndColumnsNamesPartialTable(String schemaName, String tableName, List<String> columnsName);
```
Retorna las propiedades de las columnas de la tabla *tableName* del esquema *schemaName* de la lista de columnas *columnsName* y de las claves primarias de la tabla *tableName*.

```
public List<GenericObject> getValues(String schemaName, String tableName, List<String> attributeStringList);
```
Obtiene los datos de las columnas especificadas en *attributeStringList* para la tabla *tableName*.

```
public List<AttributeColumn> getColumnsNamesTable(AttributeDatabase database, AttributeTable attributeTable) throws ExceptionKiuPlus;
```
Obtiene los nombres de las columnas de la tabla *attributeTable* de la base de datos *database*.

```
public AttributeColumn getColumnTypeParser(String schemaName, String tableName, String columnName)throws ExceptionKiuPlus;
```
Retorna un objeto del tipo *AttributeColumn* asignándole el tipo de dato de la columna *columnName* de la tabla *tableName* del esquema *schemaName*.

```
public List<GenericObject> getColumnType(String schemaName, String tableName, String columnName);
```
Obtiene el tipo de dato de la columna *columnName* de la tabla *tableName* del esquema *schemaName*.

```
public List<String> attributeListToAttributeTypeStringList(List<GenericObject> attributeList);
```
Retorna una lista de strings formados por el nombre de la columna y el tipo de datos de de las columnas en la lista *attributeList*.

Anexo III: Definición del modelo de calidad del caso de estudio.

En el anexo 3 incluimos el reporte que obtenemos de la herramienta de la definición del modelo de calidad de datos, en este caso se aplica al caso de estudio del capítulo 6.

KIUPLUS
Data quality

Modelo de calidad - Caso de estudio

Database

Name: cad
Url: jdbc:mysql://localhost:3306
System: MySQL

Instanced metrics

Caso de estudio - MI 2 - Chequear completitud

Dimension: Completeness
Factor: Density
Metrics: Density ratio
Granularity: Column
Method: Percentage Not Null Values
Operation:
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Name: percentageNotNullValues
Parameters:

Name	Type	Table	Column	Value
databaseUrl (String)	Connect to DB			
columnName (String)	Column name	llamadas	Cola	
tableName (String)	Table name	llamadas		

Caso de estudio - MI 1 - Chequear Estado

Dimension: Accuracy
Factor: Syntactic correctness
Metrics: Synt. corr. boolean
Granularity: Cell
Method: Check State
Operation:
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Name: isValidState
Parameters:

Name	Type	Table	Column	Value
data (String)	Attribute	extensiones_evento	Estado	

Caso de estudio - MI 3 - Chequear Sub Red

Dimension: Consistency
Factor: Domain integrity
Metrics: Dom. int. rule
Granularity: Cell
Method: Check prefix
Operation:
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Name: startsWith
Parameters:

Name	Type	Table	Column	Value
prefix (String)	Reference			172.31.34
data (String)	Attribute	logs	Terminal	

Caso de estudio - MI 4 - Chequear duración

Dimension: Consistency
Factor: Intra-relation integrity
Metrics: Rel. int. rule
Granularity: Cell set
Method: Check duration
Operation:
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Name: endLessStartEqualsDuration
Parameters:

Name	Type	Table	Column	Value
startDate (String)	Attribute	llamadas	HoraInicio	
endDate (String)	Attribute	llamadas	HoraFin	
duration (Double)	Attribute	llamadas	Duracion	

Caso de estudio - MI 5 - Chequear relación

Dimension: Consistency
Factor: Inter-relation integrity
Metrics: Inter. integrity rule
Granularity: Column set
Method: ForeignKey Valid Percentage
Operation:
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Name: foreignKeyValidPercentage
Parameters:

Name	Type	Table	Column	Value
table (String)	Table name	origen_eventos		
referenceTable (String)	Table name	eventos		

databaseUrl (String)	Connect to DB			
column (String)	Column name	origen_eventos	IdEvento	
referenceColumn (String)	Column name	eventos	Id	

Aggregations

Caso de estudio - Agg 1 - MI 1 - Promedio

Instanced Metric: Caso de estudio - MI 1 - Chequear Estado
End Granularity: Column
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Operation: aggAvg
Parameters:

Name	Type	Value
databaseUrl (String)	Connect to DB	
columnName (String)	Name of the results column	
tableName (String)	Name of the results table	

Caso de estudio - Agg 2 - MI 3 - Promedio

Instanced Metric: Caso de estudio - MI 3 - Chequear Sub Red
End Granularity: Column
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Operation: aggAvg
Parameters:

Name	Type	Value
databaseUrl (String)	Connect to DB	
columnName (String)	Name of the results column	
tableName (String)	Name of the results table	

Caso de estudio - Agg 3 - MI 4 - Promedio

Instanced Metric: Caso de estudio - MI 4 - Chequear duración
End Granularity: Column set
Url: http://localhost:8084/DataQualityImpCatalogoMI?wsdl
Operation: aggAvg
Parameters:

Name	Type	Value
databaseUrl (String)	Connect to DB	
columnName (String)	Name of the results column	
tableName (String)	Name of the results table	

Composed Instanced Metrics

Anexo IV: Reporte de ejecución de las métricas instanciadas del modelo de calidad datos del caso de estudio.

A continuación, incluimos el reporte que podemos obtener de la herramienta luego de ejecutar las métricas instanciadas del modelo de calidad de datos. Como en el modelo de calidad hay métricas instanciadas con granularidad celda aplicada a tablas con muchos registros el reporte es bastante extenso, incluimos las hojas representativas del reporte.

Result of Execut Modelo de calidad - Caso de estudio

Instanced Metrics

Caso de estudio - MI 5 - Chequear relación

Date of execution: 03/12/2018 18:23:03

Execution date	Result
03/12/2018 18:23:03	0.998

Caso de estudio - MI 1 - Chequear Estado

Date of execution: 03/12/2018 16:48:52

Estado	Result
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1
Archivado	1

FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
FaltaDespachar	1
Nuevo	0
Nuevo	0
Nuevo	0
TomadoSupervisor	1
TomadoSupervisor	1
TomadoSupervisor	1
TomadoSupervisor	1
TomadoSupervisor	1
TomadoSupervisor	1

Caso de estudio - MI 4 - Chequear duración

Date of execution: 03/12/2018 18:23:00

Duracion	HoraFin	HoraInicio	Result
12.0	2017-11-07 13:14:20	2017-11-07 13:14:08	1
439.0	2017-11-07 13:52:45	2017-11-07 13:48:06	0
47.0	2017-11-07 14:33:04	2017-11-07 14:32:57	0
98.0	2017-11-07 14:36:12	2017-11-07 14:35:14	0
25.0	2017-11-07 14:36:48	2017-11-07 14:36:23	1
76.0	2017-11-07 14:40:25	2017-11-07 14:39:49	0
84.0	2017-11-07 14:41:19	2017-11-07 14:40:35	0
14.0	2017-11-07 14:42:18	2017-11-07 14:42:04	1
76.0	2017-11-07 14:50:08	2017-11-07 14:49:32	0
29.0	2017-11-07 14:50:43	2017-11-07 14:50:14	1
61.0	2017-11-07 14:51:11	2017-11-07 14:50:50	0
207.0	2017-11-07 15:16:55	2017-11-07 15:14:48	0
533.0	2017-11-07 15:22:36	2017-11-07 15:17:03	0
8.0	2017-11-07 15:24:10	2017-11-07 15:24:02	1
106.0	2017-11-07 15:26:42	2017-11-07 15:25:36	0
23.0	2017-11-07 15:27:53	2017-11-07 15:27:30	1
91.0	2017-11-07 15:30:13	2017-11-07 15:29:22	0
123.0	2017-11-23 16:29:44	2017-11-23 16:28:21	0
696.0	2017-11-24 17:21:48	2017-11-24 17:14:52	0
68.0	2017-11-24 17:22:25	2017-11-24 17:21:57	0
428.0	2017-11-24 17:31:38	2017-11-24 17:27:10	0
-2.0161023173148E13	0001-01-01 00:00:00	2017-11-24 17:31:48	0
192.0	2017-11-24 17:36:36	2017-11-24 17:34:44	0
409.0	2017-11-24 17:41:41	2017-11-24 17:37:32	0
213.0	2017-11-27 12:55:29	2017-11-27 12:53:16	0

5.0	2018-03-22 15:19:50	2018-03-22 15:19:45	1
4261.0	2018-03-23 17:00:19	2018-03-23 16:57:58	0
-2.0170222182246E13	0001-01-01 00:00:00	2018-03-23 18:22:46	0
376.0	2018-03-26 14:23:24	2018-03-26 14:19:48	0
349.0	2018-03-26 14:34:06	2018-03-26 14:30:57	0
8.0	2018-03-26 14:43:16	2018-03-26 14:43:08	1
19.0	2018-03-26 14:44:27	2018-03-26 14:44:08	1
6.0	2018-03-26 14:44:52	2018-03-26 14:44:46	1
-2.017022515131E13	0001-01-01 00:00:00	2018-03-26 15:13:10	0
-2.01702251521E13	0001-01-01 00:00:00	2018-03-26 15:21:00	0
-2.0170305181504E13	0001-01-01 00:00:00	2018-04-06 18:15:04	0
1357.0	2018-04-09 16:38:02	2018-04-09 16:24:45	0
-2.0170309162915E13	0001-01-01 00:00:00	2018-04-10 16:29:15	0
9321.0	2018-04-12 16:01:23	2018-04-12 15:08:02	0
1708.0	2018-04-12 17:30:24	2018-04-12 17:13:16	0
0.0	2018-04-12 18:08:21	2018-04-12 17:53:20	0
238.0	2018-04-17 17:35:45	2018-04-17 17:33:07	0
-2.0170317130551E13	0001-01-01 00:00:00	2018-04-18 13:05:51	0
-2.0170317154408E13	0001-01-01 00:00:00	2018-04-18 15:44:08	0
173.0	2018-04-18 16:05:24	2018-04-18 16:03:51	0
-2.0170317160549E13	0001-01-01 00:00:00	2018-04-18 16:05:49	0
21.0	2018-04-18 16:16:46	2018-04-18 16:16:25	1
-2.0170317161655E13	0001-01-01 00:00:00	2018-04-18 16:16:55	0
546.0	2018-04-18 16:24:03	2018-04-18 16:18:57	0

Caso de estudio - MI 2 - Chequear completitud

Date of execution: 03/12/2018 16:42:27

Execution date	Result
02/12/2018 22:06:19	34.7023
03/12/2018 16:42:26	34.7023

Caso de estudio - MI 3 - Chequear Sub Red

Date of execution: 03/12/2018 18:21:45

Terminal	Result
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1
172.31.34.21	1

172.31.34.40	1
172.31.34.40	1
172.31.34.40	1
172.31.34.40	1
172.31.34.40	1
172.31.34.40	1
172.31.34.40	1
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
172.31.34.28	1
172.31.34.28	1
172.31.34.28	1
172.31.34.28	1
172.31.34.28	1
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0
fe80::87e:7339:c74c:d4ee%10	0

Aggregations

Caso de estudio - Agg 2 - MI 3 - Promedio

Has not results

Caso de estudio - Agg 1 - MI 1 - Promedio

Has not results

Caso de estudio - Agg 3 - MI 4 - Promedio

Has not results

Anexo V: Reporte de ejecución de las agregaciones del modelo de calidad datos del caso de estudio.

En este anexo incluimos el reporte de ejecución que se obtiene de la herramienta luego de haber ejecutadas las agregaciones seleccionando la opción de borrar los resultados de las métricas instanciadas ejecutadas previamente. A diferencia del reporte del anexo IV, vemos que para las métricas instanciadas 1, 3 y 4 no hay resultados.

KIUPLUS
Data quality

Result of Execut Modelo de calidad - Caso de estudio

Instanced Metrics

Caso de estudio - MI 1 - Chequear Estado

Has not results

Caso de estudio - MI 3 - Chequear Sub Red

Has not results

Caso de estudio - MI 2 - Chequear completitud

Date of execution: 03/12/2018 16:42:27

Execution date	Result
02/12/2018 22:06:19	34.7023
03/12/2018 16:42:26	34.7023

Caso de estudio - MI 4 - Chequear duración

Has not results

Caso de estudio - MI 5 - Chequear relación

Date of execution: 03/12/2018 18:23:03

Execution date	Result
03/12/2018 18:23:03	0.998

Aggregations

Caso de estudio - Agg 3 - MI 4 - Promedio

End granularity: Column set

Instanced metrics: Caso de estudio - MI 4 - Chequear duración

result	Instanced metric execution date	Aggregation execution date
0.2936	03/12/2018 18:23:00	03/12/2018 22:37:17

Caso de estudio - Agg 1 - MI 1 - Promedio

End granularity: Column

Instanced metrics: Caso de estudio - MI 1 - Chequear Estado

result	Instanced metric execution date	Aggregation execution date
0.999	03/12/2018 16:48:52	03/12/2018 22:37:14

Caso de estudio - Agg 2 - MI 3 - Promedio

End granularity: Column

Instanced metrics: Caso de estudio - MI 3 - Chequear Sub Red

result	Instanced metric execution date	Aggregation execution date
0.9981	03/12/2018 18:21:45	03/12/2018 22:37:15

Anexo VI: Manual de instalación

En este anexo se describen los pasos necesarios para instalar la aplicación KiuPlus y el catálogo en el servidor.

El primer paso es instalar Java SE Development Kit [37] y configurar la variable de entorno JAVA_HOME con la ubicación donde se realizó la instalación.

A continuación, instalar el servidor web Apache Tomcat, el cual se puede descargar de la página web de Apache Tomcat [34] y la guía de instalación se encuentra en [35].

Configurar la variable de entorno $CATALINA_HOME apuntando a la carpeta base de la instalación de Apache Tomcat.

Para configurar la ubicación del archivo kiuplus.properties descripto en la sección Configuración de la herramienta se debe agregar al archivo setenv.bat las líneas:

```
set “CONF_DIR=ubicacion_properties”
exit /b 0
```

Donde ubicacion_properties es la ubicación del archivo kiuplus.properties. El archivo setenv.bat se encuentra en la carpeta $CATALINA_HOME\bin.

Para hacer el deploy de la aplicación copiar el archivo KiuPlusWebAppMVC.war a la ubicación $CATALINA_HOME\webapps. Este paso se puede realizar de diversas maneras, como se indica en [36].

Se debe instalar MySQL Workbench en el mismo servidor y configurarlo con el usuario root y la contraseña admin. Luego importar el dump kiu_plus_configuracion. En caso de utilizar otro servidor para la base de datos, otro usuario o la contraseña diferente, se debe modificar el archivo hibernate.cfg.xml en Kiuplus/Logic/Resource en el código fuente y generar el war.

Por último, iniciar el servidor Apache Tomcat. Para acceder a la aplicación ir a la URL http://localhost:8080/KiuPlusWebAppMVC/.

Para instalar el catálogo copiar el archivo KiuPlusCatalog.jar al servidor y ejecutar el archivo JAR en la línea de comandos: java -jar KiuPlusCatalog.jar.