FACULTAD DE INGENIERÍA
UNIVERSIDAD DE LA REPÚBLICA

Hacia un modelo de Calidad de Datos para el Personal Software Process

INFORME PROYECTO DE GRADO

Denis Leonardo Alvez Da Costa
Juan Manuel Fernández Cardoso
Carlos María Yarza Ganem

TUTORES DEL PROYECTO
Carolina Valverde
Diego Vallespir

Montevideo, Uruguay
Noviembre, 2015

Resumen

Al aplicar un proceso de desarrollo de software, los individuos y los equipos generan datos acerca de su uso. Estos datos (de avance del proyecto e históricos) son utilizados para obtener información y estadísticas que son de gran utilidad tanto para el seguimiento del proyecto en curso como para la estimación de proyectos futuros. Entre otros datos, se requiere conocer el valor ganado del proyecto, saber cómo se encuentra la calidad del producto (por ejemplo, analizando la cantidad de defectos registrados y corregidos), realizar estimaciones de esfuerzo y costo.

Sin embargo, sucede que a veces los datos que se van recolectando durante la ejecución del proyecto de software (ya sea el tiempo dedicado en cada fase, datos sobre los defectos encontrados, datos sobre el cronograma, entre otros) tienen problemas de calidad. Si esto es así, entonces los análisis que se realicen a partir de estos datos, y por ende las decisiones que se toman, pueden no ser acertadas y llevar a los proyectos por un camino no deseado.

Dentro de los procesos de desarrollo, PSP (*Personal Software Process*) es un proceso de mejora personal que ayuda a controlar, gestionar y mejorar la forma de trabajo. El proceso está dividido en fases que se van completando mientras se desarrolla el producto de software. En cada fase, los individuos recolectan y registran datos sobre los tiempos utilizados y los defectos removidos.

Este proyecto tiene como finalidad conocer la calidad de los datos que son generados durante la aplicación del PSP. Se trata de aplicar una metodología general de análisis de calidad para este estudio particular.

Índice

Índice de cuadros

Índice de figuras

1 Introducción

El concepto de Calidad de Datos ha tomado mayor trascendencia en los últimos tiempos. Tanto en el ámbito profesional como académico, cada vez resulta más importante el nivel de calidad que poseen los datos recolectados, generados y almacenados en las bases de datos. En el ámbito profesional, en base a los datos se brindan servicios a clientes, se obtienen estadísticas y reportes a partir de los cuales se toman importantes decisiones a nivel empresarial [1]. En el ámbito académico también es necesario que la calidad de los datos que se utilizan sea la adecuada [19]. Estos pueden ser generados, por ejemplo, a partir de la ejecución de experimentos o cursos, y serán analizados para obtener determinados resultados y conclusiones. A pesar de que cada vez hay un número más importante de investigaciones y trabajos acerca del tema, se trata de una disciplina relativamente nueva dentro de la informática.

La disciplina de Calidad de Datos propone herramientas y técnicas para evaluar y mejorar la información con que se cuenta. Esta disciplina es un área de investigación en sí misma, en la cual en los últimos años se ha generado un gran volumen de trabajo enfocado a: definir los distintos aspectos de la calidad de datos, y proponer técnicas, métodos y metodologías para la medición y para el tratamiento de la calidad de los datos [14].

Por otro lado, PSP (*Personal Software Process*) [12] propone un proceso de desarrollo de software personal. Este ayuda al ingeniero a recolectar datos y a utilizarlos para mejorar su proceso de desarrollo de software, de forma de lograr una mayor productividad y una mejora en la calidad del producto desarrollado.

En cualquier desarrollo de software los equipos y los individuos generan datos acerca del uso del proceso. Por ejemplo, se registran horas de trabajo, defectos encontrados, tiempos por fase, etc. Estos datos son utilizados por las empresas a la hora de realizar el seguimiento de proyectos y evaluar el desempeño. A nivel individual PSP es un proceso de auto-mejora que ayuda a controlar, gestionar y mejorar la forma en la que se realiza el trabajo. Es un marco de trabajo estructurado, compuesto de guías y procedimientos para desarrollar software. Usado de manera adecuada PSP brinda información necesaria para hacer y poder cumplir compromisos (en términos de calidad y calendario) y hacer más eficiente y predecible la forma en que se realiza el trabajo [12]. Niveles inadecuados de calidad en estos datos puede llevar a la toma de malas decisiones por parte de la gerencia de las empresas, o a nivel personal si se quiere mejorar a partir de la experiencia individual.

En este proyecto se propone el estudio de la calidad de una base de datos que son recolectados por varios ingenieros a partir de cursos dictados desde junio de 2006 hasta setiembre de 2010, utilizando la herramienta *PSP Student Workbook [3]*. Estos cursos fueron dictados por el *Software Engineering Institute* (SEI) de la Universidad Carnegie Mellon o por asociados (*partner*) del SEI, incluyendo un número diferente de instructores en múltiples países. Analizamos los datos de 408 ingenieros que realizaron el curso.

El objetivo es conocer la calidad de los datos analizados.

1.1 Motivación

El concepto de calidad es difícil de definir, pero todos sabemos de qué se trata. La abstracción del concepto de calidad hace difícil la valoración de resultados y los beneficios que se obtienen a partir de la aplicación de diferentes técnicas y análisis de calidad.

En los últimos años esta disciplina ha ido adquiriendo mayor relevancia, convirtiéndose en un aspecto fundamental para todo sistema de información. Las organizaciones toman importantes decisiones a partir de los datos que poseen almacenados, entonces una mala calidad de los mismos puede causar importantes pérdidas económicas si las decisiones estratégicas son tomadas en base a información errónea [1].

El área de PSP no es ajena a esta temática. Si se desea mejorar el proceso que realizan los usuarios del PSP o poder alertar sobre las deficiencias de un ingeniero a la hora de construir *software*, los datos deben ser fiables de modo que las estadísticas y resultados obtenidos sean acertados. De nada serviría analizar y obtener conclusiones a partir de datos incorrectos o de mala calidad. Es más sería "nocivo" en el sentido de que representan una falsa realidad.

1.2 Objetivos

En este trabajo se plantean los siguientes objetivos:

- **Medir la calidad de los datos** generados por ingenieros que realizan cursos de PSP utilizando la herramienta *PSP Student Workbook*. Aplicar las métricas definidas y ejecutar los métodos de medición correspondientes. Presentar los resultados de la medición mediante la definición de valores de calidad.
- **Definición de un modelo inicial de calidad para el PSP:** con modelo de calidad de datos nos referimos al conjunto de dimensiones, factores, problemas de calidad y métricas definidas para este estudio. Al finalizar este primer modelo de calidad tendremos un conjunto de métricas debidamente categorizadas y agrupadas, a través de las cuales se pueden obtener los valores de calidad.
- **Generar *metadata*** (estructura y datos) que permita realizar trabajos de limpieza y migración de datos a futuro, sobre la base de datos de PSP. Se pretende dejar correctamente identificados los problemas de calidad y los errores encontrados.

1.3 Conceptos

Hay cuatro conceptos que resultan centrales en el transcurso del estudio.

El primer concepto es el de "problema de calidad". Se utiliza esta definición para referirse a un problema genérico para determinado factor y dimensión de calidad. Es una agrupación de métricas que se enfocan en un mismo aspecto de calidad o tienen la misma semántica.

El segundo concepto es el de "error", que es la instanciación de un problema de calidad sobre un atributo, tabla o un conjunto de estos. Se refiere a un error en la calidad de los datos.

El tercer concepto de "métrica de calidad", que son un instrumento que define una forma de medir un factor de calidad.

Finalmente definimos "modelo de calidad" como el conjunto de dimensiones, factores, problemas de calidad y métricas definidas para este estudio.

1.4 Trabajo realizado

Se realizó un análisis exhaustivo de la base de datos generada mediante la recolección de datos de 408 ingenieros que utilizaron la herramienta *PSP Student Workbook* para registrar diversos proyectos. Se identificaron y clasificaron problemas de calidad, de acuerdo a los conceptos de dimensiones y factores propuestos por la disciplina de Calidad de Datos.

Se utilizaron diferentes enfoques con el objetivo de definir las métricas de calidad: a partir de la herramienta, a partir de la base de datos, a partir de los *scripts* de PSP y a partir de la *grading checklist*. Aplicando estos enfoques, se registraron las posibles fuentes de problemas de calidad en los datos, y se clasificaron de acuerdo a varios conceptos: respecto a la calidad de datos (dimensiones, factores, problema de calidad, granularidad), respecto al proceso PSP (etapa, formulario, tablas y atributos afectados). Una vez que se analizaron todas las formas posibles de introducción de errores, se llegó a un conjunto de métricas de calidad.

A continuación, el conjunto de métricas definido se unió con el conjunto de métricas encontradas por el estudio [14], ya que era un proyecto de similares características sobre la misma base de datos.

Del conjunto de métricas, se clasificaron según si son factibles de que ocurran o no, ya que algunos de los problemas detectados no podrían ocurrir por controles que realiza la herramienta.

Una vez que se depuró completamente la planilla de métricas se procedió a implementar los métodos de medición mediante sentencias SQL. Se generó *metadata* en una base de datos independiente donde se almacena la información recogida por las métricas. Para ello se crearon *scripts* que implementan las métricas y *scripts* para obtener los valores de calidad en sí.

Una vez que se tienen los valores de calidad se procede a analizar estos resultados. Se hace un estudio de valores de calidad por tabla, por problema de calidad y por métrica. También se realiza un análisis transversal que contiene información sobre los usuarios con menores valores de calidad.

1.5 Contenido

A continuación se presenta el capítulo "*Calidad de datos*", que contiene una introducción a la teoría general de Calidad de Datos. Se presentan los principales conceptos manejados en el desarrollo del estudio y luego se profundiza con más detalles en las diferentes dimensiones y factores que se abordan en el proyecto. Proporciona el marco general del trabajo realizado.

Luego se presenta el capítulo "*Proceso personal de software*" para conocer sobre el tema particular del cual se quiere estudiar la calidad: PSP (*Personal Software Process*). Se describen todos los conceptos necesarios para comprender el trabajo realizado.

En el capítulo 4 se describe la metodología de identificación de problemas de calidad utilizada. En el capítulo "*Identificación de problemas de calidad sobre los datos de PSP*" se detalla cómo se realizó la identificación de posibles errores sobre la base, cómo fueron documentados y clasificados los mismos.

El siguiente capítulo "*Medición de la calidad de los datos de PSP*" presenta concretamente cómo se realiza la medición de la calidad, la definición de *metadata*, y la definición e implementación de las métricas definidas.

Luego de esto se presenta el análisis cuantitativo realizado sobre la base de datos de PSP. En el capítulo "*Resultados obtenidos*" se presentan los errores encontrados, y se agrupan los mismos de acuerdo a diversos criterios para tener diferentes enfoques. Se muestran los valores de calidad de la base de datos obtenidos en el estudio.

Finalmente se muestra un capítulo con el desarrollo de la "*Conclusión y trabajos a futuro*". También se presenta un glosario con los conceptos más importantes que se utilizan en el transcurso del análisis.

2 Calidad de datos

En este capítulo se hace una introducción al marco general de trabajo de este proyecto, que es la calidad de datos. Definiremos los principales conceptos, características e impacto en los sistemas y organizaciones, para ponernos en contexto. Se empieza por definir el concepto de calidad de datos. Se muestran diferentes clasificaciones de los tipos de datos existentes. Se definen los conceptos de dimensiones y factores de calidad, haciendo una descripción de los que se estudian en el desarrollo de este proyecto. Este capítulo se basa fuertemente en [1] y [2].

2.1 Qué es la calidad de datos

La definición más comúnmente aceptada y utilizada sobre Calidad de Datos es la de "adecuación al uso", si los datos son adecuados para un uso o contexto particular [18].

El término "Calidad de Datos" se utiliza en referencia a un conjunto de características o "dimensiones" de calidad que deben poseer los datos (tales como la exactitud, la completitud y la frescura). A menudo se confunde la calidad de datos con la exactitud de los mismos (por ejemplo si están bien escritos un apellido, una dirección, etc.). Sin embargo, la calidad de los datos es más que la exactitud, hay otras "dimensiones" como la completitud o la frescura que también son importantes con el fin de caracterizar correctamente la calidad de datos en toda su amplitud. La calidad de datos es un concepto multifacético, y en su definición hay que tener en cuenta las diferentes dimensiones de calidad.

Además de los propios datos en sí, gran parte de las metodologías de diseño para el modelo relacional abordan propiedades que afectan la calidad del esquema. Con el fin de solucionar este tipo de problemas se han propuesto varias formas de normalizar los esquemas.

Desde la perspectiva de la investigación, la calidad de los datos se ha abordado en diferentes áreas, incluidas la estadística, la gestión y la informática. Los estadísticos fueron los primeros en investigar algunos de los problemas relacionados con la calidad de los datos, al proponer una teoría matemática en datos estadísticos conjuntos a finales de la década del 60'. Luego, investigadores del área de gestión al comienzo de los 80' se centraron en la forma de controlar los datos de sistemas de fabricación con el fin de detectar y eliminar problemas de calidad de datos. Recién a partir del 1990 se empezó a investigar el tema en el área de la informática, para medir y mejorar la calidad de los datos electrónicos almacenados en bases de datos, archivos y sistemas legados.

Las diferentes dimensiones de calidad son más fáciles de detectar en ciertos casos (errores ortográficos), pero en otros se hace más complicado (datos correctos pero no admisibles, como ser una edad de 170 años, en donde el valor numérico es correcto pero no corresponde a un valor lógico).

2.2 Importancia de la calidad de datos

Los datos constituyen un recurso muy valioso para las empresas y organizaciones, al ser los mismos utilizados entre otras cosas para la toma de decisiones, siendo de suma importancia para garantizar el éxito y la supervivencia de dichas organizaciones.

Debido a esto la mala calidad de los datos puede tener consecuencias graves y de gran trascendencia para la eficacia y eficiencia de las organizaciones y empresas, así como también en la vida cotidiana, llegando en algunos casos a pérdidas millonarias.

Algunos ejemplos de estas consecuencias (en particular, en EEUU) son:

- La entrega tardía o errónea de correspondencia por parte del servicio postal debido a un error en la base de datos de direcciones, entregas erróneas de correos electrónicos o duplicación de los mismos.
- Varios registros de un mismo cliente actualizados por diferentes procesos de la organización, con inconsistencias.

- Reporte de *DW Institute*: problemas de calidad de datos le cuestan a los negocios del país, más de 600 billones de dólares al año.
- Entre 50% y 80% de registros electrónicos de criminales resultaron ser incorrectos, incompletos o ambiguos.
- En el servicio de correo, de 100000 unidades de correspondencia masiva enviada, 7000 no llego a destino debido a direcciones incorrectas.
- Más de 35% de los proyectos de TI fracasan debido a la mala calidad de datos.

Estos números demuestran el gran impacto de la mala calidad de datos en los sistemas de información.

De los ejemplos anteriores se desprende que la mala calidad de los datos puede llevar a costos elevados e innecesarios, a la baja satisfacción y desconfianza del cliente con la organización. Además de tener un alto impacto en la toma de decisiones.

Entre las fuentes de errores más comunes debido a la mala calidad de los datos se destacan:

- Sistemas de información de organizaciones públicas y privadas pueden ser vistos como un conjunto de actividades de producción separadas, almacenando su información en diferentes bases de datos, lo que a menudo caracteriza la superposición o duplicación de información (como por ejemplo registros de clientes, cuentas, direcciones, etc.).
- Muchas organizaciones establecen relaciones separadas con los miembros individuales de las familias, o en general, los grupos relacionados de personas. Luego si se quiere establecer una relación entre estas personas es muy difícil establecer correspondencias, ya que muchas veces la misma persona es ingresada en varias oportunidades.
- La integración en diferentes organizaciones de información legada con otros sistemas requiere compatibilidad e interoperabilidad entre los sistemas de información, con el nivel de base de datos necesario para garantizar la interoperabilidad física y semántica.

Los ejemplos anteriores indican la necesidad creciente de integrar información de diferentes fuentes de datos, una actividad en la que la mala calidad de los mismos obstaculiza los esfuerzos de integración. La conciencia de la importancia de mejorar la calidad de los datos es cada vez mayor en muchos contextos.

2.3 Tipos de datos

Los datos representan objetos del mundo real, en un formato que se puede almacenar y recuperar, elaborado a través de un procedimiento de software y comunicado a través de una red. El proceso de representar el mundo real por intermedio de los datos permite almacenar varios tipos de los mismos, como por ejemplo: mediciones, eventos, características de las personas, el medio ambiente, sonidos, entre otros.

Dado que los investigadores en el ámbito de la calidad de los datos deben hacer frente a un amplio espectro de posibles representaciones de datos, se han propuesto varias clasificaciones para los mismos. Luego las dimensiones y técnicas para la calidad de datos tienen que estar adaptadas para todos los tipos de datos que se describirán a continuación.

En primer lugar se pueden clasificar los datos como:

- Estructurados: cuando cada elemento de dato tiene una estructura fija asociada. Por ejemplo, tablas relacionales.
- Semi-estructurados: cuándo los datos tienen una estructura que contiene un cierto grado de flexibilidad. Por ejemplo, XML es utilizado a menudo para representar datos semi-estructurados.
- No estructurados: cuándo los datos son expresados en lenguaje natural y no se especifica estructura o tipo de dominio para definirlos.

Un segundo punto de vista ve los datos como un producto. En este modelo se puede identificar un paralelismo entre la calidad de datos y la calidad de productos. En este modelo también hay tres tipos diferentes de datos:

- *Raw data ítems*: son considerados unidades pequeñas de datos. Son utilizados para construir información y *components data ítems*. Estos tipos de datos pueden ser almacenados por largos períodos de tiempo.
- *Component data ítems*: son datos que son almacenados temporalmente hasta que el producto final es construido.
- *Information products*: son el producto de la actividad de fabricación realizada sobre los datos.

Esta clasificación de los datos como un producto, se puede utilizar para garantizar la calidad de datos en procesos de fabricación.

Una tercera clasificación, se trata de una distinción típica realizada en los sistemas de información entre datos elementales y datos agregados.

- Los datos elementales son administrados en las organizaciones por procesos operaciones y representan fenómenos atómicos del mundo real (ejemplo, número de cédula, edad, sexo, etc.).
- Los datos agregados se obtienen a partir de una colección de datos elementales mediante la aplicación de una función de agregación de ellos.

Esta clasificación es útil para distinguir los diferentes niveles de la gravedad en la medición y el logro de la calidad de los datos. Por ejemplo, la exactitud de un archivo "sexo" cambia dramáticamente si en lugar de "H" se pone por error "M" o si la edad de una persona se registra mal. En cambio, la exactitud del promedio de edad de una población de unos cuántos millones se vería mínimamente afectada.

Actualmente se maneja una nueva clasificación de datos que se desprende del entorno de las redes de computadoras y particularmente de internet:

- *Federated data*: son datos que provienen de diferentes fuentes, heterogéneas entre sí, y que luego se combinan.
- *Web data*: datos obtenidos desde la web. A pesar que se caracterizan por formatos no convencionales, son a menudo la principal fuente de información para ciertas actividades.

Las clasificaciones anteriores no se fijan en la dimensión temporal de los datos. De acuerdo a su frecuencia de cambio surge una última clasificación:

- Datos estables: son aquellos que es poco probables que se modifiquen (por ejemplo: publicaciones científicas).
- Datos que se modifican a largo plazo: son datos que tienen una frecuencia de cambio muy baja (por ejemplo: direcciones, monedas, etc.).
- Datos que cambian frecuentemente: son datos que cambian frecuentemente en un corto lapso de tiempo (como por ejemplo: información en tiempo real; tráfico, estado del tiempo, etc.).

Los datos de la base de datos bajo estudio se pueden clasificar como datos estructurados. Ya que los mismos se encuentran almacenados en tablas de una base de datos relacional. Y la otra clasificación útil para este estudio, es la clasificación temporal de los datos, que se clasifican como datos estables. Ya que la base de datos una vez que se construyó (recogiendo los datos de PSP de diversas bases), no se volvió a modificar. Los datos en ella son una recopilación de datos de otras bases.

2.4 Conceptos de calidad de datos

La calidad de datos presenta múltiples dimensiones asociadas. Cada **dimensión** captura un aspecto específico incluido en el marco general de la calidad de datos. Las dimensiones de calidad pueden referirse tanto a la extensión de los datos (valores de datos), o su intención (su esquema). Algunos ejemplos de dimensiones pueden ser: exactitud, completitud, consistencia.

Luego se definen los **factores** de calidad, que representan un aspecto particular de una dimensión. A modo de ejemplo, los factores de la dimensión exactitud son: correctitud semántica (si los datos represen-

tan entidades/estados del mundo real), correctitud sintáctica (si los datos no tienen errores sintácticos) y precisión (si los datos contienen el suficiente nivel de detalle).

Dependiendo del tipo de problema o aplicación un factor puede ser más adecuado que otro a la hora de medir o corregir la calidad para una dimensión. Una dimensión se puede ver como un conjunto de factores de calidad que tienen el mismo propósito.

También se definen **métricas** de calidad, que son un instrumento que define la forma de medir un factor de calidad. Un mismo factor de calidad puede medirse con diferentes métricas.

Finalmente se definen los **métodos** de medición, que son los procesos que implementan las métricas. Son los encargados de tomar una serie de medidas correspondientes a una métrica para una base de datos concreta, o sea, la implementación del método depende de la aplicación concreta y de la estructura de la base de datos. Una misma métrica puede ser medida por diferentes métodos.

El siguiente esquema resume la jerarquía de conceptos de calidad de datos aquí expuestos:

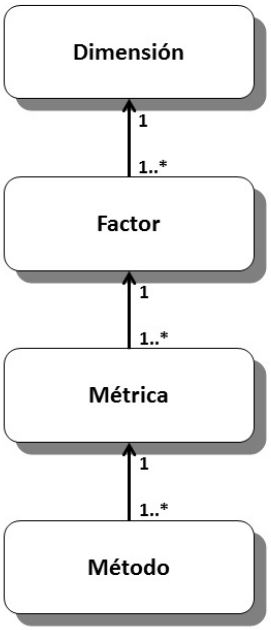

Figura 1: Relación entre los conceptos de Calidad de Datos

Como se puede ver en la Figura 1 cada dimensión está compuesta de uno o más factores, que a su vez pertenecen a una única dimensión. Cada factor está compuesto de una o más métricas, que a su vez pertenecen a un único factor. Cada métrica está compuesta de uno o más métodos, que a su vez pertenecen a una única métrica. Al estar las relaciones representadas por una composición se está indicando una relación fuerte de la siguiente manera: una dimensión no tiene sentido si no existe un conjunto de factores para representar todos sus aspectos, un factor no tiene sentido si no existen métricas para poder medirlos y una métrica no tiene sentido si no existen métodos a través de los cuales poder implementarlas.

Por último, otro de los conceptos centrales en este proyecto es el de **modelo de calidad**. Definir un modelo de calidad para un contexto dado, significa definir un conjunto de dimensiones, factores, problemas de calidad, y las relaciones existentes entre ellos, asociados a la temática en cuestión [19]. A partir de los cuales se puedan definir métricas para poder medir la calidad de los datos. El modelo de calidad definido es genérico para el contexto definido. Esto significa que si se realiza otro estudio de calidad sobre el mismo dominio de aplicación, la cantidad de métricas definidas puede llegar a variar, pero las dimensiones, factores y problemas a estudiar se mantendrían.

2.5 Dimensiones de calidad

Cada dimensión captura un aspecto específico de la calidad de los datos.
La calidad de los esquemas conceptuales y lógica es muy importante en el diseño y uso de la base de datos. A pesar de esto, se centra la atención de las definiciones de las dimensiones de calidad de datos en los valores de los mismos, los cuales se utilizan más ampliamente que los esquemas en los procesos comerciales y administrativos.

Por lo tanto, las dimensiones que se presentan a continuación se ocupan principalmente de las dimensiones de datos, aunque también se hablará de algunas de las dimensiones de esquema más relevantes.

A continuación se describirán los aspectos más importantes de algunas dimensiones de calidad, las que a nuestro entender abarcan una mayor riqueza conceptual por ser las más abordadas por los diferentes autores.

2.5.1 Exactitud (*accuracy*)

La exactitud (*accuracy*) se define como la cercanía entre un valor v y un valor v', considerando v' como la correcta representación en el mundo real del fenómeno que se pretende modelar.

Se pueden identificar tres factores para la dimensión exactitud: exactitud sintáctica, exactitud semántica y precisión.

- **Exactitud sintáctica:** es la cercanía de un valor *v* a los elementos que por definición pertenecen al dominio de *v'*. Interesa saber si *v* corresponde algún valor valido del dominio de *v'*.
  Se pueden encontrar varios problemas de exactitud sintáctica.
  - Errores de valores: por ejemplo, valores fuera de rango, errores ortográficos o errores de tipo.
  - Errores de estandarización: valores que no tienen el formato adecuado, por ejemplo un campo "sexo" con los valores 0 y 1 en lugar de "M" y "F", o unidades de medida inapropiadas como gramos en lugar de kilos.

  La exactitud sintáctica se mide por medio de funciones de comparación que evalúan la distancia entre *v* y el dominio *D* al cual pertenece *v'*.
  Exactitud semántica: es la cercanía que existe entre un valor *v* y un valor real *v'*. Mide la distancia del valor de *v* al verdadero valor del mundo real *v'*.
  Por ejemplo, tener datos sobre un cliente que referencien a una persona equivocada, o que no existe, o que contiene errores en algunos de sus atributos (como la dirección o la edad).
- **Precisión:** trata del nivel de detalle brindado por los datos en los sistemas de información.
  Por ejemplo, para un campo salario de una persona, no es lo mismo un valor de $20000 que un valor de $20454. O en una dirección, "Montevideo" y "Ejido 1430, 11200, Montevideo".

Existen diferencias apreciables entre la exactitud semántica y la exactitud sintáctica. Por ejemplo si bien es razonable medir la exactitud sintáctica utilizando una función de distancia, la exactitud semántica se mide mejor con un valor de tipo (SI, NO) o (CORRECTO, INCORRECTO). En consecuencia la exactitud semántica coincide con el concepto de corrección.

Contrariamente a lo que sucede con la exactitud sintáctica, para medir la exactitud semántica de un valor *v* es necesario saber el valor correspondiente del mundo real.

De los argumentos anteriores se deduce que la exactitud semántica es por lo general más compleja de medir que la exactitud sintáctica. Cuando se sabe a priori que la tasa de errores es baja, y los errores resultan típicamente en errores tipográficos, entonces la exactitud sintáctica tiende a coincidir con la exactitud semántica.

La exactitud sintáctica puede lograrse mediante la sustitución de un valor inexacto con el valor más cercano en el dominio de definición, bajo el supuesto de que es el correcto.

2.5.2 Completitud (completeness)

La completitud se puede definir como el grado en que los datos son de suficiente amplitud, profundidad y alcance para la tarea en cuestión.

Intuitivamente la completitud caracteriza el grado en que está representado el mundo real correspondiente. La completitud en el modelo relacional es caracterizada por dos factores:

- **Densidad**: interesa medir cuánta información tengo y cuánta me faltan sobre las entidades del sistema de información. Trata la presencia y significado de los valores nulos.
  En un modelo con valores nulos, la presencia de un valor nulo tiene el significado de un valor faltante, es decir, un valor que existe en el mundo real pero por alguna razón no está disponible o no se conoce.
- **Cobertura:** interesa saber cuántas entidades de la realidad contiene mi sistema de información. Se trabaja bajo la validez de uno de los dos supuestos: suposición de mundo abierto y suposición de mundo cerrado. La suposición de mundo cerrado establece que solo los valores realmente presentes en una tabla relacional representan hechos del mundo real y que no hay otros. En cambio en el supuesto de mundo abierto no podemos afirmar ni la verdad ni la falsedad de los hechos que no están representados en las tuplas del modelo.

2.5.3 Consistencia (consistency)

Esta dimensión captura las violaciones de las reglas semánticas definidas sobre un conjunto de los elementos de datos, como pueden ser tuplas en tablas relacionales o registros de archivos. En referencia a la teoría relacional, las restricciones de integridad son una ejemplificación de tales reglas semánticas.
Las restricciones de integridad son propiedades que deben ser satisfechas por todas las instancias en un esquema de base de datos.

Es posible distinguir dos grandes categorías de restricciones, que corresponden a los factores de la dimensión Consistencia:

- Restricciones intrarrelación: pueden considerar atributos individuales o varios atributos de una misma relación. Por ejemplo, supongamos un esquema de relación de empleados, con los atributos: nombre, apellido, edad, antigüedad y salario. Una restricción de integridad intrarrelación de un solo atributo puede ser: la edad está comprendida entre 0 y 100. Y un ejemplo de integridad intrarrelación de varios atributos puede ser: si la antigüedad es menor que tres entonces el salario anual no puede ser mayor a los 250.000 pesos anuales.
- Restricciones interrelación: implican atributos de más de una relación.

La mayor parte de restricciones de integridad son consideradas dependencias. Se pueden considerar los siguientes tipos de dependencias:

- *Key*: este es el tipo más simple de dependencia (**integridad relacional**). Dada una instancia de relación *r* definida sobre un conjunto de atributos, es decir, un subconjunto *k* de los atributos tienen dependencia de clave en *r*, si no hay dos filas de *r* que tengan los mismos *k*-valores.
- Inclusión: este tipo de dependencia es un tipo de restricción muy común, también se conoce como **integridad referencial**. Una dependencia de inclusión en una instancia de relación *r* indica que algunas columnas de *r* están contenidas en otras columnas de *r*. Una restricción de clave externa es un ejemplo de dependencia de inclusión, que indica que las columnas se refieren a una relación que deben estar contenidas en las columnas de clave primaria de la relación que se hace referencia.

- Funcional: dada una instancia relacional *r*, sean X e Y dos conjuntos no vacíos de atributos de *r*, *r* satisface la dependencia funcional X -> Y, si para cada par de tuplas *t1* y *t2* en *r* se cumple: si t1.X = t2.X a continuación t1.Y = t2.Y. Donde la notación t1.X significa la proyección de la tupla t1 en los atributos de X.

2.5.4 Frescura (freshness)

Un aspecto importante de los datos es su cambio y actualización con el transcurso del tiempo. Anteriormente se habló de una clasificación de tipos de datos de acuerdo a la dimensión temporal, en términos de datos estables y de largo plazo que no se modifican con frecuencia, y datos que cambian frecuentemente. Los principales factores relacionados con la frescura son: actualidad (*currency)*, edad *(timeliness) y* volatilidad *(volatility)*.

- **Actualidad**: refiere a que tan rápidamente se actualizan los datos, o que tan vigentes son los mismos. Un sistema de información es una vista de entidades/estados de la realidad en un momento dado, cuando la realidad correspondiente cambia el sistema puede quedar desactualizado (por ejemplo, las direcciones, teléfonos de los clientes pueden cambiar). La actualidad es típicamente medida con respecto a la última actualización los datos. Para los tipos de datos que cambian con una frecuencia fija, la última actualización de los datos nos permite calcular la medida directamente. Por el contrario para los tipos de datos cuyo cambio de frecuencia puede variar, una posibilidad es calcular una frecuencia de cambio promedio y realizar el cálculo de actualidad con respecto a ella, pero sabiendo que se pueden dar errores.
- **Volatilidad**: caracteriza la frecuencia con que los datos varían en el tiempo. Por ejemplo los datos estables como las fechas de nacimiento tienen volatilidad igual a 0 ya que no varían en absoluto. Sin embargo datos como la cotización en bolsa de acciones, un tipo de datos que varían con frecuencia, tienen un alto grado de volatilidad ya que estos datos son válidos para períodos muy cortos de tiempo.

  La volatilidad es un factor que caracteriza inherentemente a ciertos tipos de datos. Una métrica para volatilidad viene dada por la longitud de tiempo en que los datos siguen siendo válidos.
- **Edad**: expresa que tan útiles son los datos actuales para la tarea en cuestión. La dimensión edad esta movida por el hecho de que es posible tener datos actuales que son realmente inútiles porque es tarde para su uso particular. Por ejemplo el calendario de cursos de una universidad puede ser actual, contiene los datos más recientes, pero no es oportuno si el mismo está disponible después del inicio de los cursos.

  La edad implica que los datos no solo son actuales, sino también que son adecuados en el tiempo a los eventos que los utilizan. Por lo tanto una posible medida consiste en una medición y verificación de que los datos estén disponibles antes de su tiempo de uso previsto.

2.5.5 Unicidad (uniqueness)

Esta dimensión trata la unicidad de las entidades en los sistemas de información. Se desprenden dos factores, la duplicación y la contradicción.

La **duplicación** se produce cuándo una entidad del mundo real se almacena dos o más veces en una fuente de datos. Estos problemas se pueden apreciar en estructuras de almacenamientos como archivos u otras estructuras que no manejan el concepto de clave. En lo que refiere a bases de datos este problema estaría limitado por la utilización de claves de las tablas.

Un ejemplo de dos registros duplicados sería: <4.367.330-1, “J.M. Fernández”, 27 años, Masculino>
<4.367.330-1, “J.M. Fernández”, NULL, Masculino>.

Donde los valores de la clave y los demás atributos coinciden (o son nulos algunos).

La **contradicción** se da cuándo una entidad además de estar repetida se encuentra con contradicciones de una representación a otra. Por ejemplo, campos con diferentes valores en las entidades en cuestión. Un ejemplo de contradicción sería: <4.367.330-1,"J.M. Fernández",27 años, Masculino>

<4.367.330-1,"Juan Fernández",25 años, Masculino > .

Donde los valores de la clave pueden coincidir o no, y hay diferencias en algunos de los otros atributos.

2.6 Limpieza de datos

La limpieza de datos tiene como objetivo la detección y eliminación de errores e inconsistencias de los datos con el fin de mejorar la calidad de los mismos.

Debido a la amplia variedad de posibles inconsistencias en los datos y el volumen de los mismos, la limpieza de datos es considerada como uno de los mayores problemas en el almacenamiento de los mismos.

Un gran número de herramientas de funcionalidad variable está disponible para apoyar estas tareas. Sin embargo, una parte significativa del trabajo de limpieza y transformación tiene que ser a menudo hecho manualmente o por programas de bajo nivel que son difíciles de implementar y mantener.

Un **método** de limpieza de datos debe satisfacer varios requisitos. Los principales son que debe detectar y eliminar todos los errores e inconsistencias tanto en las fuentes individuales de datos como en la integración de múltiples fuentes.

Mientras que la gran mayoría de los investigadores han abordado la traducción de esquemas e integración de los mismos, la depuración de los datos ha recibido poca atención en la comunidad científica.

2.6.1 Problemas de limpieza de datos

Como veremos, todos los problemas de limpieza de datos están estrechamente relacionados y por lo tanto deben ser tratados de una manera uniforme. Las transformaciones de datos son necesarias para apoyar cualquier cambio en la estructura, la representación o el contenido de los datos.

Estas transformaciones se hacen necesarias en muchas situaciones. Tal es el caso de la evolución del esquema, la migración de un sistema existente a un nuevo sistema de información, o la integración de múltiples fuentes de datos.

Los problemas de limpieza de datos se dividen básicamente en dos grandes grupos:

- Limpieza de una sola fuente.
- Limpieza de múltiples fuentes.

Para el estudio en particular a realizar en este proyecto nos enfocaremos en el tipo de problemas de limpieza de una sola fuente ya que la base de datos bajo estudio es de esas características.

2.6.2 Problemas de una sola fuente

La calidad de una fuente de datos depende en gran medida del grado en el que esta se rige por el esquema y restricciones de integridad que controlan los valores de datos permitidos. Para las fuentes sin esquema, como archivos, hay pocas restricciones en cuanto al registro y almacenamiento de datos, dando lugar a una alta probabilidad de errores e inconsistencias. Los sistemas de base de datos, por otra parte, hacen cumplir las restricciones de un modelo de datos específico, así como la aplicación específica de restricciones de integridad. Por ejemplo, el enfoque relacional requiere valores de los atributos simples, integridad referencial, etc.

Los problemas de calidad de datos relacionados con los esquemas se producen debido a la falta de adecuación de los modelos específicos de la aplicación a las restricciones de integridad. Esto se puede deber a las limitaciones del modelo de datos a nivel de diseño, o esquemas pobres, o a la falta de restricciones de integridad definidas para limitar la sobrecarga para los controles de integridad.

Los problemas específicos de la instancia, por otra parte, se refieren a los errores e inconsistencias que no se pueden prevenir a nivel de esquema (por ejemplo, errores ortográficos).

Teniendo en cuenta que la limpieza de las fuentes de datos es un proceso costoso, la prevención de datos "sucios" es un paso importante para reducir el problema de limpieza. Esto consiste entre otras cosas, en un diseño apropiado del esquema de base de datos y restricciones de integridad, así como de las solicitudes de entrada de datos.

2.6.3 Fases de limpieza de datos

En general, la limpieza de datos implica varias fases que se describen a continuación:

- ***Análisis de los datos***: Para detectar qué tipos de errores e inconsistencias se van a extraer, se requiere un análisis detallado de datos. Además de una inspección manual de los datos o muestras de datos, se deben utilizar programas de análisis para obtener las propiedades de los metadatos y detectar problemas de calidad en los mismos.
- ***Definición del proceso de transformación y reglas de mapeo***: En función del número de fuentes de datos, su grado de heterogeneidad y la "suciedad" de los datos, puede ser necesario tener que ejecutar un gran número de transformaciones de datos y etapas de limpieza. En los primeros pasos de limpieza de datos se pueden corregir problemas de las instancias de una sola fuente, y preparar los datos para la integración. Los pasos posteriores tratan la integración de datos y los problemas de limpieza para múltiples fuentes, como son los duplicados.
- ***Verificación:*** La exactitud y la eficacia de un flujo de trabajo y las definiciones de transformación deben ser probados y evaluados, ya sea en una muestra o una copia de los datos de origen, para mejorar las definiciones si es necesario. Se pueden necesitar varias iteraciones de los pasos de análisis, diseño y verificación (por ejemplo, algunos errores sólo se manifiestan después de aplicar ciertas transformaciones).
- ***Transformación:*** La ejecución de los pasos de la transformación se realiza ya sea mediante la ejecución del flujo de trabajo de ETL (Extraction, Transformation and Load) para la carga y actualización de un datawarehouse, o realizando consultas desde múltiples fuentes.
- ***Reflujo de datos limpios:*** Una vez que los errores se eliminan (en una sola fuente), los datos deben ser limpiados, reemplazando los datos "sucios" en las fuentes originales para evitar volver a hacer el trabajo de limpieza para futuras extracciones de datos.

2.6.4 Resolución de conflictos

Se debe especificar y ejecutar un conjunto de pasos para las transformaciones para resolver los diversos problemas de calidad en los esquemas y a nivel de instancia. Existen varios tipos de transformaciones que se realizarán sobre las fuentes de datos individuales con el fin de hacer frente a los problemas de una sola fuente y para prepararse para la integración con múltiples fuentes. Estos pasos son:

- **Extracción de valores de atributos de forma libre**: se refiere a capturar múltiples valores individuales que deben extraerse para lograr una representación más precisa y apoyar nuevas medidas de limpieza tales como instancias correspondientes y eliminar duplicados. Ejemplos típicos son campos de nombre y dirección. Las transformaciones necesarias en esta etapa son la reordenación de los valores dentro de un campo para hacer frente a transposiciones de palabras, y la extracción del valor para dividir el atributo.
- **Validación y corrección**: Este paso examina cada instancia de origen de los errores de entrada de datos y trata de corregir de forma automática en la medida de lo posible. La corrección ortográfica basada en un diccionario de búsqueda es útil para identificar y corregir las faltas de ortografía. Por otra parte, los diccionarios de nombres geográficos y códigos postales ayudan a corre-

gir los datos de dirección. Dependencias de atributos (fecha de nacimiento - edad, precio total - Precio de unidad / cantidad, ciudad - prefijo telefónico, etc.) pueden ser utilizados para detectar problemas y sustituir los valores perdidos o valores erróneos correctos.

- **Estandarización**: Para facilitar la adaptación e integración, los valores de los atributos deben ser convertidos a un formato coherente y uniforme. Por ejemplo, las entradas de fecha y hora deben ser llevados a un formato específico, nombres y otros datos de cadena deben ser convertidos a mayúsculas o minúsculas, etc. Por otra parte, las abreviaturas y los sistemas de codificación deben resolverse mediante la consulta de diccionarios de sinónimos especiales o la aplicación de reglas de conversiones predefinidas.

2.7 Herramientas de apoyo

Una gran variedad de herramientas está disponible en el mercado para apoyar la transformación de datos y las tareas de limpieza.

Algunas herramientas se concentran en un dominio específico, tales como la limpieza de nombres y direcciones, o una fase de limpieza específica (análisis de los datos o la eliminación de duplicados). Debido a su dominio restringido, las herramientas especializadas suelen realizar estas tareas muy bien, pero deben complementarse con otras herramientas para hacer frente a la amplia gama de problemas de transformación y limpieza. Otras herramientas, por ejemplo de ETL, proporcionan transformación integral y capacidades de flujo de trabajo para cubrir una gran parte de la transformación de los datos y el proceso de limpieza. Un problema general de las herramientas de ETL es su limitada interoperabilidad debido a las interfaces de programación de aplicaciones propietarias (API) y formatos de metadatos de propiedad por lo que es difícil combinar la funcionalidad de varias de estas herramientas.

En este trabajo no se utilizarán herramientas de apoyo para la medición o limpieza de datos.

3 Proceso personal de software

3.1 Introducción

Vivimos en una era donde la tecnología juega un papel fundamental en la vida cotidiana de las personas. En la actualidad la sociedad interactúa con una gran cantidad de productos tecnológicos, algunos tan pequeños como un reloj, un teléfono celular, como también con otros de mayor tamaño, como por ejemplo, los cajeros automáticos, computadoras, barcos, aviones, entre muchos otros. Y lo que tienen en común todos ellos es que requieren de un software para que puedan funcionar.

La tecnología crece a un ritmo tan acelerado que los programas se vuelven cada vez más grandes y complejos. Los desarrolladores de software deben ser conscientes de que para lograr buena calidad en sus productos, deben tener el control de las etapas del proceso.

Si no se sabe qué se está haciendo, es difícil mejorar la manera de hacerlo. Cuando se sigue un proceso definido, el desarrollador es más consciente de cómo y en que se está trabajando. Para eso es fundamental que se lleve un registro detallado de todas las actividades que se realizan en las etapas del proceso.

Si la información recolectada es utilizada de manera adecuada, es altamente probable que la calidad de sus productos a futuro sea cada vez mejor, como también aumenta enormemente las probabilidades de realizar mejores estimaciones y planificaciones de los proyectos.

Un reporte presentado por *Standish Group* [4], muestra que el porcentaje de proyectos de software exitosos es muy bajo. Es por eso que la ingeniería de software juega un rol fundamental, ya que gracias a ella se realizan diversos estudios e investigaciones, con el fin de mejorar estos resultados.

Uno de sus principales objetivos es encontrar procesos y metodologías, que sean sistemáticas y predecibles a fin de mejorar la calidad y desarrollo del producto de software.

En la década del 1980, en el Instituto de Ingeniería en Software (En Inglés, *Software Engineering Institute* o SEI) de la Universidad de Carnegie Mellon, Watts Humphrey desarrolló un proceso de mejora personal llamado *Personal Software Process* (de ahora en adelante se abrevia con las siglas PSP). El principal propósito es mejorar las habilidades del ingeniero de software, brindando formularios, guías y procedimientos para el desarrollo del software. También ayuda a mejorar sus planificaciones, a realizar un seguimiento más preciso de su propio rendimiento y a medir la calidad de los productos desarrollados. Además de permitir identificar dónde y cómo mejorar.

3.2 El camino de PSP

Para aprender a usar y aplicar este proceso personal se dictan cursos, en los cuales se presentan un conjunto de ejercicios que deben ser resueltos programáticamente por los inscriptos, con el fin de conocer y entender las seis versiones ascendentes que define PSP.

Se comienza con la versión inicial del proceso identificada como PSP 0 hasta madurar con el proceso completo, o sea hasta alcanzar PSP 2.1. En la Figura 2, se muestra la evolución de las versiones. PSP 0 es un proceso simple y definido, que establece que el desarrollo de los programas se realice de manera habitual, con la restricción adicional de que se debe registrar tiempos y defectos encontrados. La siguiente versión es PSP 0.1, extiende a PSP 0 mediante una adición de la medición del tamaño. En PSP 1 se enseña a usar el historial de los datos de tamaño y tiempo, para estimar el tamaño del programa y tiempo que dura el desarrollo. Una vez estimado, en la etapa PSP 1.1 se planifica las tareas y se las calendariza. En PSP 2 se realiza las revisiones de diseño y código, y por último con PSP 2.1 se crean las plantillas de diseño con el fin de reducir los defectos inyectados durante el diseño de sus productos.

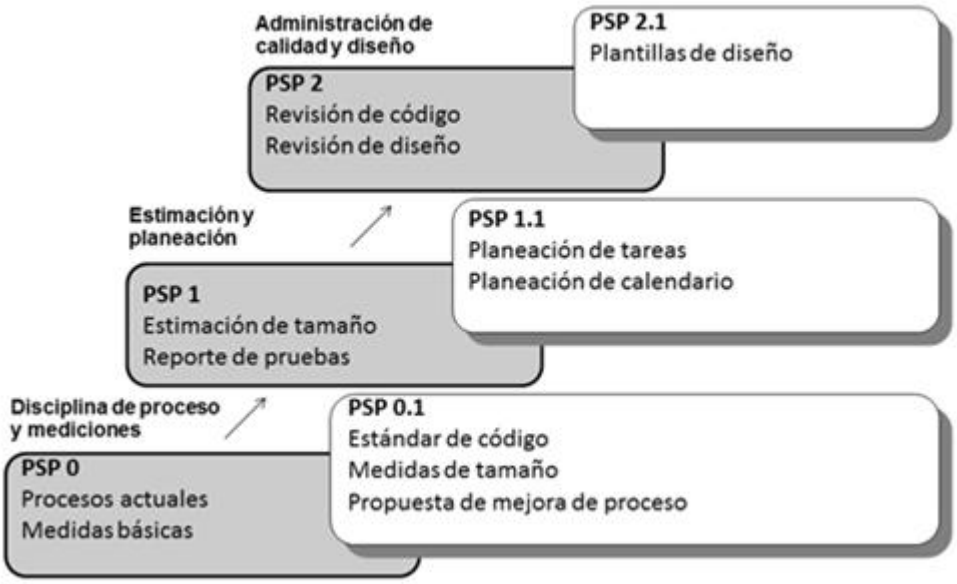

Figura 2: Evolución de PSP

3.3 PSP 0

El principal objetivo de PSP 0 es proporcionar un contexto para desarrollar el primer programa con PSP y recolectar la información del proyecto. El flujo del proceso PSP 0 se muestra en la Figura 3. Los ***scripts*** son una guía que define cada uno de los pasos del proceso, los registros ayudan a guardar los datos del proceso, y el resumen del plan permite registrar y reportar los resultados.

El proceso PSP 0 brinda los siguientes beneficios:

- Una estructura conveniente para realización de tareas a pequeña escala.
- Un marco para medir esas tareas.
- Una base para la mejora del proceso.

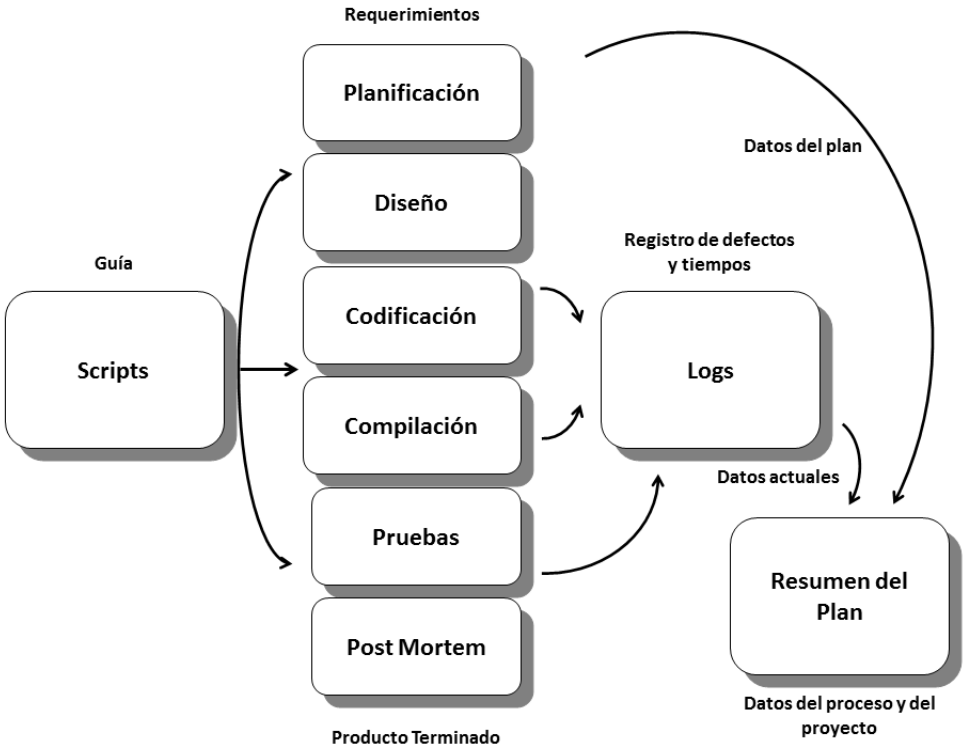

Figura 3: Flujo proceso PSP 0

3.3.1 ELEMENTOS DEL PROCESO PSP

En la Figura 3, se presentan todos los elementos del flujo del proceso PSP 0. En la fase de planificación, se realiza el plan de proyecto. Luego le sigue la fase de desarrollo que incluye las siguientes cuatro etapas: diseño, codificación, compilación y pruebas (testeo). Por último, la fase de *postmortem*, que es donde se compara la performance actual, con todo lo planificado anteriormente y se realiza un reporte resumido.

Otro elemento son los *scripts*, estos sirven como guía a través de todas las versiones del proceso.

En PSP 0, existen cuatro tipos de *scripts*:

- Script de proceso.
- Script de planificación.
- Script de desarrollo.
- Scripts de *postmortem*.

Por más detalles sobre los *scripts*, ver sección 1.1 del ANEXO C *"Datos complementarios de psp"*.

3.3.2 MEDIDAS PSP 0

PSP 0 utiliza dos medidas:

1. El tiempo.
2. Cantidad de defectos.

La finalidad de registrar datos del tiempo y defectos, es ayudar a realizar el plan y gestionar los proyectos. Estos datos muestran dónde se dedica el tiempo, como también dónde se inyectan y se corrigen los defectos. Estos datos sirven para el desarrollador a evaluar su rendimiento personal, además de indicarle dónde debe intensificar esfuerzos con el fin de mejorar la productividad y calidad de los productos.

Registrando el tiempo

Se registran los siguientes datos:

- El proyecto o programa que se está trabajando.
- La fase del proceso en cual se encuentra.
- La fecha y hora de comienzo y fin de trabajo en la tarea.
- Tiempo de cualquier interrupción.
- Tiempo neto o delta trabajado en la tarea.
- Comentarios.

Registrando defectos

Cuando se detecta un defecto y se decide corregirlo, se debe contabilizar el tiempo que llevó la corrección del mismo. Una vez solucionado, se debe ingresar toda la información para ese defecto. La mayoría de las herramientas automáticamente asignan un único número para el identificar el defecto, pero hay que ingresar manualmente el tipo del defecto, las fases en cuales fue inyectado y removido, el tiempo de corrección, y una breve descripción del defecto. Si se genera un defecto corrigiendo otro, se registra como referencia el número del defecto relacionado.

Resumen del plan de proyecto de PSP 0

Para realizar la planificación del proyecto se utiliza el formulario de resumen del plan de proyecto que define PSP, un ejemplo de este se puede apreciar en el Cuadro 1. En PSP 0, el tiempo estimado para el desarrollo del proyecto se ingresa en la herramienta como el tiempo total planificado. Al no contar con datos sobre la distribución del tiempo de desarrollo por fase, no se puede asignar el total del tiempo a través de las fases de PSP para este proyecto.

Si se registran apropiadamente todos los tiempos y defectos, la herramienta de PSP automáticamente completa los valores reales por fase en el resumen del plan de proyecto, generando el tiempo total real que se dedicó al proyecto.

Durante la fase *postmortem*, se examinan los datos en el formulario completo del plan de proyecto. En la columna "hasta la fecha" del formulario se mantiene los tiempos y defectos reales de todos los programa. Esta columna muestra la distribución de los datos hasta la fecha a través de las fases.

Ejemplo del resumen del plan de proyecto para PSP 0

Estudiante:	Estudiante 3	**Fecha:**	ene-19
Programa:	Desviación Estándar	**Nro. Programa:**	1
Instructor:	Roberto Perez	**Lenguaje:**	C++

Tiempo por fase	**Plan**	**Real**	**Hasta la Fecha**	**% Hasta la Fecha**
Planificación		5	5	4.3
Diseño		30	30	25.6
Codificación		32	32	27.4
Compilación		15	15	12.8
Pruebas		5	5	4.3
Post Mortem		30	30	25.6
Total	180	117	117	100.0

Defectos Inyectados		Real	Hasta la Fecha	% Hasta la Fecha
Planificación		0	0	0.0
Diseño		2	2	28.6
Codificación		5	5	71.4
Compilación		0	0	0.0
Pruebas		0	0	0.0
Total		7	7	100.0

Defectos removidos		Real	Hasta la Fecha	% Hasta la Fecha
Planificación		0	0	0.0
Diseño		0	0	0.0
Codificación		0	0	0.0
Compilación		6	6	85.7
Pruebas		1	1	14.3
Total Desarrollado		7	7	100.0
Después del Desarrollo		0	0	

Cuadro 1: Resumen del plan de proyecto

3.4 PSP 0.1

Es la continuación de PSP 0, en el cual se agrega lo siguiente:

- Medición y estimación de tamaño de los programas que se desarrollan.
- Se utilizan estándares de codificación.
- Se generan las propuestas de mejoras del proceso (PIPs - *Process Improvement Proposals*) que más se adecuen a la mejora personal.

Los principales objetivos de esta versión son:

- Medir el tamaño de los programas producidos.
- Realizar el conteo de tamaño para los programas producidos.
- Realizar estimaciones de tamaño precisas y más exactas.

3.4.1 MIDIENDO EL TAMAÑO DEL SOFTWARE

El tamaño de software es la unidad mediante la cual representamos la magnitud del producto. Usualmente el software se puede medir en Líneas de Código (LOC), casos de usos, controles en pantalla, objetos de base de datos, etc.

Para realizar el plan de desarrollo de software, en una primera instancia hay que realizar el diseño conceptual, siendo el mismo una representación de alto nivel de la solución que busca mostrar los elementos de diseño fundamentales y sus funciones. Luego a partir de este, se debe estimar el tamaño del producto a construir. Con la estimación del tamaño, se puede estimar el tiempo que puede llevar para desarrollar el producto. Para poder estimar el tamaño del software es necesario tener una forma de medirlo.

Usando la medida del tamaño para la planificación

Si el tamaño de un programa es aproximadamente proporcional al tiempo requerido para desarrollarlo, entonces los datos del tamaño y tiempo se dicen que están correlacionados. La correlación es el grado en que dos conjuntos de datos están relacionados. El valor de la correlación varía entre -1.0 a 1.0. Cuanto más cerca de 1.0 se encuentre se dice que los dos conjuntos de datos, llamados *x* e *y*, son altamente correlacionados. Esto significa que al incrementar el valor de *x* se incremente proporcionalmente el valor de *y*. Por el contrario, cuanto más cerca de -1.0 se encuentre, al incrementar el valor x, el valor de *y* decremen-

ta. En PSP para que el valor de la correlación sea útil para la estimación y planificación debe ser mayor que 0.7.

Es importante saber si la correlación es significativa, esto se obtiene utilizando lo que es denominado *significance*. Mide la probabilidad de que la relación podría haber ocurrido por casualidad. Por lo tanto, una *significance* de 0.25 indica que una cuarta parte de las veces podría haber ocurrido como resultado de fluctuaciones aleatorias en los datos. Este valor puede ser calculado manualmente y se consideran como adecuados los que son menores que 0.05.

Encontrando la correlación y el valor de *significance* para los datos del tamaño y tiempo correspondientes un número de programas, se puede determinar si la medida del tamaño es útil para predecir el tiempo de desarrollo.

Los métodos de estimación en PSP son igualmente usables con medidas de tamaño tales como elementos de la base de datos, páginas de documentos, formularios, LOC (líneas de código) o cualquier otra medida que cumpla con los criterios de medición de tamaño y se correlacione con el esfuerzo de desarrollo.

3.4.2 CONTEO DE TAMAÑO

Incluso después de la selección de la definición de la medida del tamaño, hay muchas maneras de usarla. Principalmente cuando se trabaja en un desarrollo en equipo, donde se producen múltiples versiones de un mismo producto. Para hacer un seguimiento de todos los cambios y adiciones del programa, y para reunir todos los datos requeridos para la estimación, es necesario usar un sistema de conteo para el tamaño.

Los elementos básicos en un sistema de conteo de tamaño son los siguientes:

- **Base:** Esta es una medida base del programa sin modificar a la que se le agregarán mejoras. Para un nuevo programa la base es 0.
- **Adicional:** Este es el nuevo código que es agregado a la base.
- **Modificado:** El código modificado es una parte del código base que ha cambiado.
- **Eliminado:** El código eliminado es una parte del código base que es removido y no usado en el próximo programa o nueva versión de este.
- **Reutilizado:** Cuando un programa existente, o una rutina de programa es copiada de alguna librería o de otra manera incluida en el nuevo programa, es contado como código reutilizado solamente si no se ha modificado.
- **Adicional y Modificado:** Para muchos tipos de desarrollo de software, el esfuerzo requerido para escribir una nueva línea de código es raramente comparable con el esfuerzo requerido para modificar una existente. En estos casos, es conveniente combinar las medidas de tamaño adicional y modificado en una medida sola "Adicional y Modificado". Esta medida es generalmente usada en los ejercicios de estimación de esfuerzo de desarrollo en PSP y para medir y gestionar la calidad.
- **Nuevo Reutilizable:** Una estrategia común de desarrollo es construir una librería con las funciones más usadas y reusarlas cada vez que sea necesario. La medida de tamaño clasificada como "Nuevo Reutilizable" es el código nuevo de desarrollo destinados para incluir en una librería reutilizable.
- **Total:** La medida total de tamaño del programa completo, independientemente de cómo fueron producidas sus partes.

3.4.3 USANDO LOS DATOS DEL TAMAÑO

Los datos del tamaño son de gran utilidad para la planificación, gestión de la calidad y análisis del proceso.

Utilizar la medida del tamaño para la planificación

En la planificación del trabajo a desarrollar, si está definida la medida del tamaño y se posee los datos del histórico del tamaño y tiempo, entonces se puede usar estos datos para estimar con precisión el tamaño del nuevo producto. Con estos mismos datos, se puede estimar el esfuerzo del desarrollo.

Evaluar la calidad del programa

En la evaluación de la calidad de un programa, es muy útil comparar entre productos. Por ejemplo, al dividir el número de defectos encontrados en una fase por el tamaño del programa se obtiene la densidad de defectos para esa fase. Esto es importante para la planificación de los proyectos porque los costos de las pruebas, mantenimiento y servicio generalmente están relacionados con los defectos del programa. Con datos de programas similares, se puede utilizar la estimación de la densidad de defectos y tamaño para estimar los costos de las pruebas, mantenimiento y servicios.

3.4.4 CALCULANDO LA PRODUCTIVIDAD

La productividad es generalmente una medida del esfuerzo en horas requeridos para realizar una unidad de trabajo. Si bien es un simple cálculo, no es tan fácil utilizar los datos correctos y apropiados para su interpretación. Para calcularla hay que dividir el tamaño del producto producido por el total de horas dedicadas a su producción.

3.5 PSP 1

Al utilizar PSP 0.1 se logra registrar no solo los datos del tiempo y defectos, sino que también los datos del tamaño de los programas desarrollados. Esto es determinante para poder aplicar PSP 1 dado que para realizar las estimaciones de tamaño y tiempo se utilizan todos estos datos del histórico. Le agrega estimaciones fundamentadas de tamaño, ya que en las etapas anteriores eran estimaciones intuitivas y basadas en la experiencia. También incluye estimaciones de recursos y un reporte de las pruebas.

3.5.1 PLANIFICACIÓN

PSP se enfoca en el proceso de planificación personal y de los productos que se producen. A pesar que la planificación personal, del equipo de trabajo y de gerencia de proyecto están todas relacionadas. Una planificación detallada es la clave para una planificación precisa, aprendiendo a hacer planes precisos para proyectos chicos, es el primer paso esencial para aprender hacerlos para proyectos grandes.

En la Figura 4 se puede apreciar el marco en que se realiza la planificación. Las tareas a desarrollar se muestran con rectángulos, mientras que los datos, reportes y productos se presentan con círculos. Comienza con las necesidades del cliente, el primer paso es definir los requerimientos. Luego para relacionar el plan con el producto a construir, hay que generar el diseño conceptual. A partir de este se puede estimar el tamaño del nuevo producto. Con la estimación, usar el histórico de los datos para estimar el tiempo que llevara todo el trabajo. Finalmente comenzando con la fecha de inicio de la planificación del proyecto, se propaga las horas de trabajo disponibles sobre el calendario del proyecto para producir el cronograma. El último paso es calcular la fecha que esperamos que finalice el trabajo.

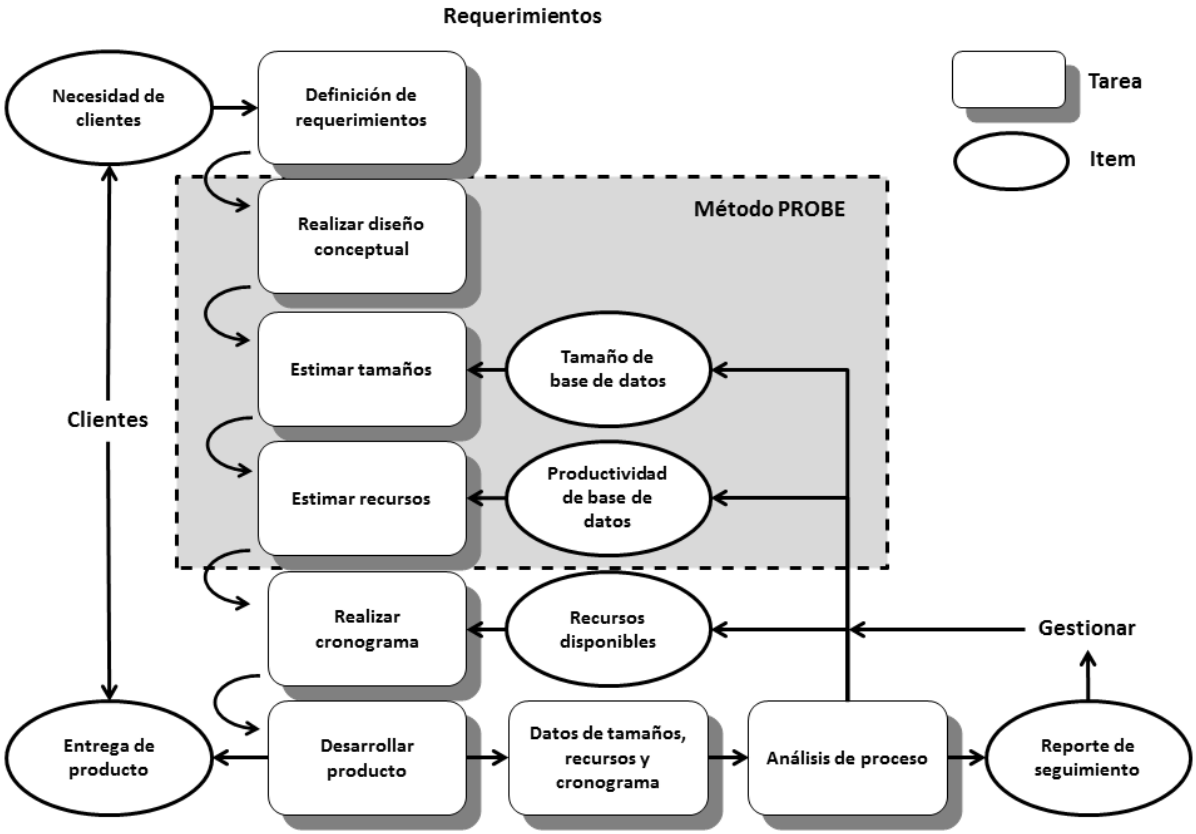

Figura 4: Planificación

3.5.2 ESTIMACIÓN DE SOFTWARE

En principio, la estimación se realiza comparando el trabajo planificado con trabajos previos. Mediante la división de la planificación del producto en partes más pequeñas y comparando cada parte con datos sobre partes similares anteriores, se puede estimar el tamaño del nuevo producto. Es necesario contar con los datos de los productos que se han desarrollado y del trabajo requerido. Se requiere de un método para poder utilizar los datos del histórico y realizar la estimación.

Con PSP, primero se hace la estimación del tamaño y después se estima el tiempo de desarrollo. Para que la estimación del tamaño sea precisa, debe basarse en el diseño conceptual inicial, y este diseño deberá reflejar la forma en que se planificó construir el producto.

Lo que se necesita es algún *proxy* que relacione el tamaño del producto a las funciones que se visualizan. Un *proxy* es un sustituto que puede ayudar a determinar el tamaño más probable para el producto. Ejemplo de *proxy* son clases, tablas, campos, o pantallas.

Utilizar *proxy* para estimar

Para utilizar *proxy* para estimar, hay que dividir los datos históricos del tamaño en categorías y rangos de tamaño.

Para estimar el tamaño se debe agrupar los datos dentro de categorías funcionales. Luego estimar cuantas partes de cada categoría son necesarias para el producto, y el tamaño relativo de cada parte. Finalmente decidir cómo se relaciona cada parte con su categoría, ejemplo muy pequeño, pequeño, medio, grande, o muy grande.

Resumiendo para estimar el tamaño, primero hay que producir el diseño conceptual e identificar sus partes. Luego decidir sobre la categoría funcional para cada parte y cuántos elementos probablemente contenga. Por último determinar si cada parte pertenece dentro del rango de tamaño muy pequeño, pe-

queño, medio, grande, o muy grande. De la tabla de tamaños relativos, se puede estimar el tamaño de cada parte y calcular el tamaño total estimado para del programa.

Método de estimación PROBE (*PROxy-Based Estimating*)

El método PROBE ayuda a obtener los datos necesarios para realizar la estimación, a usar estos datos para hacer estimaciones, a medir y mejorar la precisión de las estimaciones.

Los pasos básicos del procedimiento de estimación del tamaño se describe a continuación:

Paso 1: Diseño conceptual

Luego de completar el diseño conceptual, se identifican las partes del producto que se planea desarrollar. En este momento se está preparado para comenzar la estimación del tamaño.

Paso 2: Estimación del tamaño de las partes

Para estimar el tamaño de las partes encontradas en el diseño conceptual, hay que determinar el tipo de las partes y determinar la cantidad de elementos (métodos) de cada parte y el tamaño relativo de cada una. Una vez que se conozca si el elemento es muy pequeño, pequeño, medio, grande o muy grande, se obtiene el tamaño del elemento según la tabla que contiene los tamaños en relación categoría-tamaño relativo.

Categoría	Muy pequeño	Pequeño	Medio	Grande	Muy grande
Calculation	2.34	5.13	11.25	24.66	54.04
Data	2.60	4.79	8.84	16.31	30.09
I/O	9.01	12.06	16.15	21.62	28.93
Logic	7.55	10.98	15.98	23.25	33.83
Set-Up	3.88	5.04	6.56	8.53	11.09
Text	3.75	8.00	17.07	36.41	77.66

Cuadro 2: Ejemplo que contiene la cantidad de LOC por categorías, para clases de C++

Paso 3: Estimación del tamaño de las partes reutilizables y adicionales a la base

Si se encuentra una parte disponible que provea las funcionalidades requeridas por el diseño conceptual, entonces se debe ser capaz de reutilizarla. Siempre que estas partes funcionen según lo previsto y posean una calidad apropiada, al reutilizarlas se puede ahorrar un tiempo considerable. PROBE considera dos tipos de partes reutilizables. Primero las partes que se toman de una librería reutilizable. Segundo, las nuevas partes reutilizable, estas son algunas partes adicionales a las que ya se ha estimado.

Por último se agrega las estimaciones de todas las partes para obtener el tamaño estimado del *proxy* (E). Este valor es utilizado para realizar las proyecciones de tamaño y tiempo.

Paso 4: Procedimiento de estimación del tamaño

En este paso se verifica los datos para determinar qué método aplicar (A, B, C o D). El método que brinda una estimación más precisa es el método A, pero tiene más exigencias y además hay que contar con datos históricos.

Paso 5: Procedimiento de estimación del tiempo

Nuevamente se verifica los datos para determinar sí se puede utilizar el método A para las estimaciones del tiempo. A pesar que el método A es el más preferido para las estimaciones del tiempo, se puede usar como alternativa los métodos B, C y D.

Paso 6: El intervalo de predicción

Los cálculos finales en la plantilla de estimación del tamaño son para el intervalo predictivo. Es un rango determinado estadísticamente para la estimación de tiempo y tamaño, dentro del cual es probable que caiga el valor real.

3.6 PSP 1.1

Con PSP 1.0 se es capaz de registrar los datos de tiempo, defectos y tamaño. También mediante la utilización del método PROBE se puede estimar el tamaño y tiempo de desarrollo. Hasta el momento, luego de finalizada la estimación, se comenzaba directamente con el desarrollo sin tener el control de cómo se está con respecto a lo planificado. PSP 1.1 es útil para poder realizar el seguimiento en todo momento, analizar el progreso y mitigar los riesgos. Permite utilizar los resultados del proceso de estimación para producir un plan de proyecto.

Se identifican tres objetivos importantes del proceso PSP 1.1:

- Planificar los recursos.
- Planificar el calendario.
- Seguimiento del desempeño frente a lo planificado.

3.6.1 REALIZACIÓN DE LA PLANIFICACIÓN

A continuación se detallan los siete pasos necesarios para realizar la planificación:

1. **Estimar los recursos:** Comenzar estimando los recursos, esto sería el total de horas estimadas para el proyecto.
2. **Estimar las tareas:** Se podría utilizar el "% a la fecha" para asignar el tiempo entre las fases del proyecto. Para proyectos grandes, se deben utilizar datos de tareas similares ya realizadas para hallar la estimación del tiempo de las nuevas tareas. Luego ingresar en la plantilla de planificación de tareas, en la columna de plan de horas, el tiempo de cada una de las tareas.

Fase	Tarea	Plan Horas	Plan Horas Acumulado	PV	PV Acumulado	Plan Semanas	Horas Reales	Semana Real	EV	EV Acumulado
Plan	Book Plan	15.10	15.10	0.93	0.93	1	13.8	2	0.93	0.93
Draft	Preface	8.14	23.24	0.50	1.43	2	35.3	3	0.50	1.43
Draft	Chapter 1 Draft	21.70	44.94	1.32	2.75	3	28.6	10	1.32	10.93
Draft	Chapter 2 Draft	44.75	89.69	2.75	5.50	6	62.5	10	2.75	13.68
Draft	Chapter 3 Draft	55.60	145.29	3.42	8.92	9	41.2	13	3.42	20.52
Draft	Chapter 4 Draft	44.75	190.04	2.75	11.67	12	27.3	14	2.75	23.27
Draft	Chapter 5 Draft	56.46	256.49	4.09	15.76	15				
Draft	Chapter 6 Draft	56.45	322.94	4.09	19.85	18	67.6	7	4.09	9.61
Draft	Planning Drafts	0	322.94	0	21.23	18				
Draft	Chapter 7 Draft	55.60	378.54	3.42	23.27	21				
Draft	Chapter 8 Draft	66.45	444.99	4.09	27.36	24				
Draft	Chapter 9 Draft	32.55	477.54	1.99	29.35	26				
Draft	Quality Drafts	0	477.54	0	31.40	26				
Draft	Chapter 10 Draft	55.60	522.29	2.75	32.10	28				
Draft	Chapter 11 Draft	66.45	588.74	4.09	36.19	31	57.5	5	4.09	5.52
Draft	Chapter 12 Draft	66.45	655.19	4.09	40.28	37				
Draft	Design Drafts	0	655.19	0	43.08	37				

Cuadro 3: Ejemplo de una plantilla de planificación de tareas con datos ingresados.

3. **Estimar las horas de las tareas semanales:** Excepto para el primer proyecto con PSP, utilizar los datos del histórico como guía para estimar las horas de las tareas.
4. **Calcular las horas disponibles:** Generalmente los desarrolladores simultáneamente trabajan en más de un proyecto a la vez, por ese motivo a veces resulta un poco difícil estimar cual es la cantidad de horas que se destinará a trabajar en cada proyecto. Luego de tener en cuenta todos los compromisos y trabajos extras al momento de realizar la estimación, se debe ingresar en la plantilla de planificación de calendario, el tiempo que se dedicará al proyecto en cada una de las semanas.

Fecha	Semana	Plan Horas	Horas Acumuladas	Horas Reales Acumuladas	PV	PV Acumulado	EV	EV Acumulado
3/8	1	20	20	12.60	12.6	0.93	0	0
3/15	2	20	40	33.47	46.07	0.50	0.93	0.93
3/22	3	20	60	25.70	71.77	1.32	0.50	1.43
3/29	4	10	70	16.58	88.35	0	0	1.43
4/5	5	15	85	22.05	110.40	0	4.09	5.52
4/12	6	15	100	32.93	143.33	2.75	0	5.52
4/19	7	20	120	32.08	175.42	0	4.09	9.61
4/26	8	15	135	25.80	201.22	0	0	9.61
5/3	9	20	155	34.58	235.80	3.42	0	9.61
5/10	10	15	170	31.65	267.45	0	4.07	13.68
5/17	11	20	190	29.65	297.10	0	0	13.68
5/24	12	20	210	31.95	329.05	2.75	0	13.68
5/31	13	20	230	46.68	375.73	0	6.84	20.52
6/7	14	20	250	31.88	407.61	0	2.75	23.27
6/14	15	20	270			4.09		
6/21	16	20	290			0		
6/28	17	20	310			0		

Cuadro 4: Ejemplo de una plantilla de planificación de calendario con datos ingresados.

5. **Dependencias de las tareas del plan:** Organizar las tareas en el orden en que se espera hacerlas.
6. **Hacer la planificación:** Comenzar con la primera tarea, verificar las horas planificadas acumuladas. Analizando este dato se puede saber en qué semana termina dicha tarea, luego este valor se ingresa en la columna de la planificación de la semanas. Analizar tarea por tarea hasta completar toda la planificación.
7. **Definir los hitos:** Finalmente para facilitar el seguimiento y reporte del proyecto, se identifican los hitos claves y se listan con sus respectivas fechas de finalización previstas.

3.6.2 VALOR GANADO

Cuando se planifica un proyecto con muchas tareas, se necesita una forma de poder realizar el seguimiento y reporte del progreso, particularmente cuando se completan las tareas en diferente orden que el originalmente planeado. Si todas las tareas tomaran la misma cantidad de tiempo o si fueran solamente una pocas tareas, no sería muy difícil, pero típicamente en los proyectos hay muchas tareas de diferentes tipos y tamaños. En PSP, la medida del valor ganado (EV – *Earned Value*) proporciona una forma conveniente para gestionar los problemas de seguimientos y reportes. El método EV establece un valor para cada tarea, independientemente de su tipo. Cada vez que se completa una tarea, se gana esta cantidad EV. Para juzgar el progreso con relación al plan, simplemente comparar cada valor EV de cada tarea para cada semana, con el valor ganado planificado, o PV, para esa semana. Si el acumulativo EV es igual o excede el acumulativo de PV, entonces se está por delante de lo planificado.

Para calcular PV, se debe determinar el porcentaje estimado de cada tarea, con la relación entre la cantidad de horas que esta demanda en relación al total de horas del proyecto entero. Realmente se gana EV cuando la tarea es completada.

El valor ganado realmente es útil cuando los cambios de planes son frecuentes. También es una forma precisa para medir el estado del proyecto y estimar su fecha de finalización.

3.7 PSP 2

En esta fase se incorporan la revisión de diseño y código, con el objetivo de mejorar la calidad y productividad de los productos generados. Ambos son importantes, tanto por la forma en que se realizan, por cómo se utilizan los datos personales para la mejora de las habilidades de revisión del ingeniero.
Al realizar la revisión del diseño y código se mejora la calidad y productividad del trabajo. Además revisando el trabajo propio, se debería revisar todo lo que se utiliza como base para este trabajo. Esto incluye los requerimientos del programa, diseño a alto nivel y los detalles de diseño. Luego de implementado el programa, se revisa antes de compilar o probar. Si se encuentran problemas, arreglar los defectos, corregir la lógica defectuosa, simplificar la estructura del programa, y llevar el código al estándar personal y del equipo. Si el programa es confuso o no está muy claro, entonces agregar comentarios o reescribir para hacerlo más simple y entendible. Para no tener re-trabajo se debe desarrollar los programas de tal manera que sean fáciles de leer y de entender.

Se encontraran más detalles respecto a este tema, en la sección 3.4 ANEXO C.

3.8 PSP 2.1

En este proceso los objetivos son similares a los de PSP 2, ya que ambos tienen como meta principal y en común, la administración de la calidad. Pero la diferencia radica en que en PSP 2.1, todo se centra sobre el diseño del proyecto, brindando un criterio que le permita al ingeniero determinar si la etapa de diseño está completa o no. Si el diseño contempla todo lo requerido o si aún faltan artefactos importantes por analizar y revisar.

Una vez alcanzado estas metas, podemos decir que el ingeniero está listo para avanzar siguiente nivel, y puede sentirse seguro de la calidad del producto final desarrollado.
En este punto se mencionó una breve descripción de PSP 2.1, debido a que excede al tema tratado en este documento.

3.9 Herramienta PSP *Student Workbook*

PSP *Student Workbook* es una herramienta Access creada por el SEI, con el fin de facilitar el trabajo de los usuarios, y brindarles un soporte para que puedan trabajar con PSP de una forma más cómoda y eficiente. La herramienta facilita enormemente el trabajo de los usuarios que la utilizan, ya que cuenta con *scripts*, formularios, medidas, realiza cálculos complejos, permite hacer planificaciones, llevar un rastreo de los proyectos desarrollados, permite gestionar la calidad, realizar análisis profundos, acceder al histórico de datos, provee una serie de materiales que pueden resultarles muy útiles a los usuarios, entre otros.

Con PSP *Student Workbook* se pueden realizar proyectos en cualquiera de las versiones de PSP (0, 0.1, 1.0, 1.1, 2.0 y 2.1), teniendo luego la opción de completar los formularios correspondientes a la versión elegida.

Por más detalles respecto a la herramienta PSP *Student Workbook*, acceder a la sección 3.5 del ANEXO C.

4 Identificación de problemas de calidad sobre los datos de PSP

En este capítulo se presenta la forma de trabajo utilizada para definir las métricas de calidad e identificar los problemas de calidad sobre los datos de PSP.

En la sección de "*Metodología de trabajo*" se presentan los elementos conceptuales en los cuales se basó el análisis de calidad de datos, así como también los resultados de dicho análisis. Se define el concepto de *problema de calidad*, uno de los principales conceptos utilizados en el transcurso del proyecto.

En la sección de "*Definición y análisis de métricas de calidad de datos*" se define la forma de trabajo utilizada y los enfoques empleados para la definición de las métricas.

Luego en la sección de "*Problemas de calidad de datos definidos*" se detallan los problemas de calidad de datos a alto nivel. En la siguiente sección de "*Métricas de calidad de datos definidas*" se muestran las métricas de calidad identificadas.

En la sección "*Mapeo de métricas de calidad con estudio externo*", se detalla cómo se llevó a cabo la unión de las métricas identificadas en este proyecto con las métricas ya existentes de otro proyecto sobre los mismos datos, indicando los motivos sobre dichas diferencias.

Finalmente en la sección "*Modelo de calidad de datos para PSP*" se define el modelo inicial de calidad de datos para PSP.

4.1 Metodología de trabajo

Para llevar a cabo este trabajo se estudiaron los conceptos de calidad de datos, las diferentes dimensiones y factores de calidad, y se identificaron qué errores pueden presentarse sobre los datos en el contexto de procesos de desarrollo de software. Luego se instanciaron dichos problemas de calidad a nuestro proyecto en particular, cuyo objetivo es conocer, analizar y evaluar la calidad de los datos obtenidos a partir del uso de la herramienta *PSP Student Workbook* por más de una persona.

Los problemas de calidad de datos se generan cuando, al utilizar los datos, estos no tienen los valores adecuados de calidad [14]. En este proyecto, utilizamos el concepto de **problema de calidad** para definir conceptualmente un problema genérico para determinado factor y dimensión de calidad. En este sentido, un problema de calidad puede ser visto como la agrupación de métricas que se enfocan en un mismo aspecto de calidad, o tienen la misma semántica (por ejemplo: métricas que buscan valores nulos en diferentes campos se pueden agrupar bajo el problema de calidad "registro incompleto"). Para un factor y dimensión de calidad pueden existir varios problemas de calidad diferentes. A la instanciación de un problema de calidad sobre un atributo, tabla o un conjunto de estos, le llamamos **error**. Es un error en la calidad de los datos.

También se adquirieron conocimientos sobre PSP (*Personal Software Process*), las diferentes etapas y los diferentes formularios que hay que completar, los cálculos y estimaciones que realiza. Es importante conocer la realidad y el contexto bajo estudio de forma de identificar cuáles problemas de calidad será necesario considerar sobre sus datos.

Se exploró la herramienta *Access "PSP Student Workbook"*, sus diferentes versiones y formularios por versión. A partir de ella fue posible deducir los diferentes problemas de calidad relacionados con los factores y dimensiones estudiados para este proyecto. Es a través del estudio de la herramienta que se puede conocer los problemas de calidad que introducen los diferentes usuarios a la hora de su uso.

También se estudió la base de datos utilizada por la herramienta *Access*, para comprender el mapeo de los campos de los formularios con los atributos de las tablas de la base, y así poder identificar problemas no encontrados a través de la herramienta.

A continuación se detallan los diferentes métodos utilizados para identificar los problemas de calidad, y cómo fueron clasificados en los factores y las dimensiones estudiadas. Las dimensiones y factores que se abarcan en este proyecto son los descritos en la sección 2 *"Calidad de datos"* de este informe, en

la cual se describe de manera general sus características. Luego se detalla la definición de las métricas relacionadas con los problemas de calidad encontrados en el ANEXO A *"Métricas PSP Calidad de Datos 2015"*.

Finalmente se establece una unión con los problemas de calidad identificados por el trabajo: *"Un estudio de la calidad de los datos recolectados durante el uso del Personal Software Process (Carolina Valverde, Fernanda Grazioli, Diego Vallespir). Facultad de Ingeniería – Universidad de la República – 2012"* [14].

Esto se hace debido a que se trata de un estudio de Calidad sobre la misma base de datos. Al unir los resultados de los dos estudios logramos un estudio más completo desde el punto de vista de la calidad, complementando las métricas identificadas. Este estudio se detalla en la sección 4.4 "*Mapeo de métricas de calidad con estudio externo"*.

4.2 Definición y análisis de métricas de calidad de datos

Utilizamos 2 enfoques para la identificación de las métricas de calidad de datos. Por una parte, aplicamos un enfoque teórico más general, que parte de los factores y dimensiones propuestos por la disciplina de calidad de datos e instancia el enfoque teórico de la calidad de datos a nuestro proyecto en particular.
Por otra parte, utilizamos un enfoque más particular sobre la temática que estamos estudiando (PSP). Para este segundo enfoque hay que tener en cuenta la teoría de PSP (cálculos, estimaciones, medidas, etc.), la herramienta PSP que se utiliza para el ingreso de datos, los *scripts* y *checklist* del curso de PSP.

Ambos enfoques se aplican sobre la base de datos de la herramienta, ya que será sobre estos datos que se analizará la calidad.

De estos enfoques se desprenden las siguientes formas de trabajo que se utilizaron para definir las métricas de calidad de datos.

- **Exploratorio**: basado en la herramienta PSP *Access* y sus diferentes formularios, así como también en la base de datos de la herramienta.
  En esta forma de trabajo se hizo énfasis en los cálculos de PROBE, y campos de otros formularios que requieren algún tipo de cálculo o estimación. Se realizó una recorrida por todos los formularios de las diferentes versiones de PSP de la herramienta buscando posibles inconsistencias. También se revisó el esquema de la base de datos de la herramienta, buscando relaciones entre las tablas y entre los atributos.
- **A partir de los factores y dimensiones de calidad de datos**: basado en la herramienta *PSP Access* con sus formularios y en la base de datos de dicha herramienta.
  Esta forma de trabajo se centró principalmente en las propiedades de los atributos particulares de las tablas y las posibles relaciones entre ellos.
  Complementa a la primera ya que ambas tratan de abarcar aspectos complementarios de la calidad de datos. En este caso se excluyeron todos los cálculos que se realizan en la parte de PROBE, ya que fueron incluidos en el punto anterior.
  Al igual que en la búsqueda exploratoria se realizó una recorrida por todos los formularios de las diferentes versiones de la herramienta buscando problemas de calidad sobre los datos. Para buscar problemas de integridad se recorrieron las diferentes tablas de la base de datos, buscando relaciones entre los atributos.
- **A partir de los *scripts***: basado en los *scripts* que son proporcionados para realizar el curso de PSP.
  Esta forma de trabajo constituye un elemento diferente a los anteriores, por lo tanto son una nueva fuente para identificar posibles problemas de calidad.

A diferencia del enfoque exploratorio, en este punto se realiza una búsqueda de problemas de calidad, pero siguiendo el orden establecido en el *script* correspondiente a la versión de PSP actual, de forma de respetar cada uno de los puntos especificados en el mismo.
Para este enfoque se utiliza solamente la herramienta, navegando por los distintos formularios de forma ordenada, según lo indicado en el *script*.

- **A partir de las *grading checklist***: basada en las *grading checklist*.
  De esta forma se encontraron posibles problemas de calidad no identificados con las búsquedas anteriores.
  En este enfoque se realiza una serie de verificaciones puntuales, los cuales están agrupadas según los formularios de la herramienta, correspondientes a la etapa de PSP en que se encuentre.
  Este enfoque también limita la identificación solamente por medio de la herramienta. Además es un complemento del enfoque anterior, ya que a partir de los *scripts* se tiene un orden para proceder y navegar por los distintos formularios de la herramienta, mientras que con los *grading checklist*, se intenta prevenir que los usuarios cometan errores puntuales en los mismos.

En la Figura 5 se muestra la relación entre los elementos con los que se contó para hacer el análisis y las formas de trabajo.

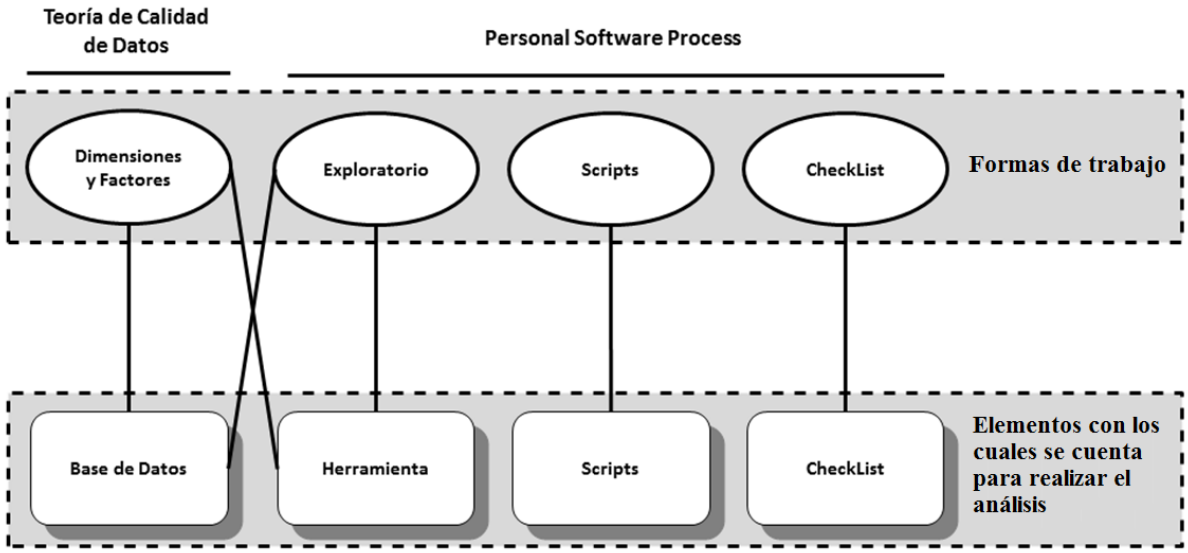

Figura 5: Elementos con los que se cuenta y formas de trabajo

Como se puede observar en la Figura 5 se realizó la identificación de métricas de calidad para encontrar problemas de calidad de datos a partir de todos los elementos disponibles: la herramienta *Access PSP* y su correspondiente Base de Datos, los *scripts* y la *grading checklist*, siempre apoyándonos en los conceptos teóricos de Calidad de Datos.

A partir de las cuatro formas de trabajo se construyó una planilla (ANEXO A*)* en la que se pueden ver todas las **métricas** encontradas, y detalles específicos de las mismas.

Dicha planilla tiene el siguiente formato:

- **Id Métrica Estudio [14]**: mapeo con las métricas del estudio [14].
- **Factibilidad**: indica si el error que detecta es factible que ocurra o no.
- #: Identificador de la métrica de calidad.
- **Versión**: indica la etapa de PSP en la cual se identificó el problema de calidad a partir de la métrica. Esto no significa que el problema no pueda presentarse en otra etapa de PSP.

- **Formulario**: indica el formulario de la herramienta en la cual se identificó el problema de calidad a partir de la métrica.
- **Definición**: descripción general de la métrica de calidad.
- **Semántica**: es una descripción más detallada de la métrica de calidad. Especifica cómo se aplica la métrica y el resultado esperado.
- **Unidad de medición de resultado**: tipo de resultado (por ej.: *boolean*) que se obtiene como resultado de ejecutar la métrica de calidad.
- **Granularidad**: corresponde al nivel al que se aplica cada métrica de calidad. La granularidad puede ser a nivel de celda, tupla, atributo e incluso la base entera.
- **Dimensión**: a la cual está asociada la métrica de calidad.
- **Factor**: al que está asociada la métrica de calidad.
- **Tablas involucradas**: que involucra la métrica de calidad.
- **Tabla a evaluar**: tabla principal que recorre la métrica.
- **Campos clave de la tabla a evaluar**: clave primaria de la tabla a evaluar.
- **Atributos**: de las tablas que involucra la métrica de calidad.
- **Método de medición**: forma de implementación utilizada para ejecutar la métrica de calidad (ejemplo: consulta SQL).
- **Problema de calidad**: corresponde al problema de calidad asociado a la métrica de calidad.
- **Forma de trabajo para la búsqueda**: se refiere a la forma de trabajo que se utilizó para identificar la métrica de calidad. Las mismas fueron definidas anteriormente.
- **Definidas sobre datos con errores de migración:** los datos que involucra esa métrica contienen errores de migración (valor del campo: SI/NO).
- **Clasificación:** posibles valores: error, sospechoso y mejora (se detalla en la sección 6.2).

Una vez completado el registro de las métricas identificadas con los cuatro métodos descritos anteriormente, se procedió a hacer la unión con las encontradas en el estudio [14]. Para esto, se procedió a comparar las métricas de calidad propuestas por ambos estudios para encontrar las coincidencias. Una vez identificadas aquellas métricas que no habían sido consideradas en nuestro estudio, se agregaron al total de métricas encontradas por nosotros, justificando el motivo de su inclusión en cada caso. En secciones posteriores se detallan los resultados obtenidos (cantidad de coincidencias y cantidad de métricas adicionadas a nuestro proyecto).

4.3 Problemas de calidad de datos definidos

Como se mencionó anteriormente el estudio de calidad se basa en las siguientes dimensiones y factores de calidad de datos:

Dimensión	Factor	Problema de calidad
Exactitud	Correctitud sintáctica	Valor fuera de rango
	Precisión	Precisión incorrecta
Completitud	Densidad	Registro incompleto
	Cobertura	Registro inexistente
Consistencia	Integridad de dominio	Reglas de integridad de dominio
	Integridad intra-relación	Reglas de integridad intra-relación
	Integridad referencial	Reglas de integridad referencial
Unicidad	Duplicación	Registro duplicado
	Contradicción	Registro contradictorio

Cuadro 5: Relación dimensión-factor-problema de calidad

A continuación se definen los problemas de calidad identificados para cada factor:

4.3.1 Exactitud

- **Correctitud sintáctica**
  - *Valor fuera de rango*: contempla los atributos cuyos valores tienen que estar dentro de un rango establecido.
    En este caso se tienen pocas métricas y en todas ellas se buscan valores de ciertos campos fuera del rango dado por tres desviaciones estándar de la media. O sea, se trata de encontrar valores que no se encuentren en el rango *[max(average – 3*standard_dev, 1), average + 3* standard_dev]*, donde "*average*" es la media y "*standard_dev*" es la desviación estándar.
- **Precisión**
  - *Precisión incorrecta*: abarca todos los atributos que requieren ser representados con cierta precisión o con determinada estructura, pero no fueron ingresados con el detalle esperado.
    Este estudio se refiere concretamente a datos de tipo *DateTime* que tendrían que tener una representación de la forma "DD/MM/AAAA HH:MM", pero el dato está almacenado con el formato "DD/MM/AAAA", en el cual no se puede apreciar la precisión de la hora. Esto puede llevar en ciertos casos a estimaciones erróneas de tiempos, ya que la interpretación de esa hora vacía puede ser diferente.

4.3.2 Completitud

- **Densidad**
  - *Registro incompleto*: este problema de calidad identifica registros incompletos. Por registro incompleto nos referimos a valores faltantes en algún campo o campos que no se encuentran en la realidad modelada. La falta de información en muchos casos se puede considerar como un error, una omisión, o la falta real del dato. Se define este problema de calidad para poder agrupar estos casos.
    La forma utilizada para medir este problema de calidad, consiste en verificar si los valores de los campos involucrados que correspondan son nulos o vacíos.
    Es uno de los problemas de calidad que tiene más métricas asociadas. Las mismas han sido encontradas a través de todos los métodos de búsqueda.
    La granularidad es por lo general a nivel de celda, ya que lo que se pretende es verificar campos por separado. También se definen algunas métricas cuya granularidad es a nivel de tupla, ya que se involucra más de un atributo de la tabla en cuestión.
- **Cobertura**
  - *Registro inexistente*: contempla los registros que existen en la realidad pero no se encuentran almacenados en los datos bajo estudio. Por ejemplo, no puede existir un proyecto finalizado para el cual no se hayan completado todas las fases.
    Se trata de detectar un conjunto de datos faltantes, los cuales son necesarios para representar la realidad modelada.
    En el caso de este problema de calidad son pocas las métricas encontradas, pero son todas del estilo de que si existen ciertos registros en una tabla, deben existir otros en tablas diferentes que complementen la información.

4.3.3 Consistencia

- **Integridad de Dominio**
  - *Reglas de Integridad de Dominio*: considera todos los problemas que surgen de no respetar las restricciones de dominio. Las restricciones del dominio del dato son las que definen si el dato es correcto o no. Estas reglas definen el rango, la estructura o el posible dominio de valores de los mismos.

    Es uno de los problemas con mayor cantidad de métricas encontradas. Esto se debe a su extenso alcance, ya que es un problema con granularidad a nivel de celda. Lo que se trata de encontrar con este problema son valores de atributos que no cumplan las condiciones dadas por el dominio del mismo. Uno de los ejemplos más comunes es encontrar un valor numérico que no se encuentre en el rango de valores válidos predefinidos por la lógica del atributo (enumerados). Otro ejemplo es encontrar un valor tipo texto pero que se encuentre fuera de los valores predefinidos por la lógica del sistema.
- **Integridad Intra-Relación**
  - *Reglas de Integridad Intra-Relación*: considera todos los problemas que surgen de no respetar las reglas que establecen la relación entre los atributos de una misma tabla.

    Es un problema con muchas métricas asociadas. Por ejemplo, la fecha de inicio de un programa debe ser menor que la fecha de finalización.
- **Integridad Referencial**
  - *Reglas de Integridad Referencial*: abarca todos los errores y falta de control de integridad referencial que puede haber en los datos. También controla que se cumplan las reglas existentes entre los valores de atributos de dos o más tablas.

    Si bien se supone que una base de datos debe controlar la integridad referencial de claves, al tratarse de varias bases de datos *Access*, que luego se integraron en una base de datos *MySQL*, es conveniente realizar este control.

    El control entre atributos que no son claves es similar al control de integridad intra-relación, pero en lugar de controlar valores de atributos de una misma tupla, se controlan valores de atributos de tuplas pertenecientes a diferentes tablas. Aplican los mismos ejemplos que los vistos para integridad intra-relación.

4.3.4 Unicidad

- **Duplicación**
  - *Registro Duplicado:* corresponde a los errores de duplicación de registros. Se consideran tanto duplicados exactos como duplicados a nivel conceptual (no son exactamente iguales, pero se reconoce que identifica al mismo objeto de la realidad).

    Para este problema se trata de encontrar registros asociados a ciertas entidades en los que los valores de ciertos atributos coincidan. No se busca la coincidencia total, si no que se verifican ciertos atributos que permitan afirmar que se trata de la misma entidad.
- **Contradicción**
  - *Registro Contradictorio*: considera los registros que si bien representan el mismo objeto de la realidad, poseen contradicciones en sus datos. Por ejemplo, los registros de log de tiempo de una fase deben ser únicos.

    En las métricas asociadas a este problema se trata de identificar las entidades que conceptualmente son la misma, pero difieren en algunos valores de sus campos. Por eso se dice que los registros que tienen esas características son contradictorios.

La forma de identificar este problema de calidad, es buscar conceptos que tuvieran el mismo valor en ciertos campos, pero que difieran en otros.

Como se puede observar el Cuadro 5, no se considera la dimensión de calidad frescura ni sus factores. Esto se debe a que si bien se consideró dicha dimensión en nuestro estudio, no se encontraron posibles problemas de calidad relacionados.

4.4 Métricas de calidad de datos definidas

En el Cuadro 6 se detallan la cantidad de métricas identificadas por forma de trabajo aplicada.

Forma de trabajo	Total métricas	Factibles	No factibles
Dimensión-factor	48	27	21
Exploratorio	149	55	94
Scripts	144	72	72
Doc-Estimating Review Times	4	4	0
Grading Checklist	37	37	0
Total	**382**	**195**	**187**
Total sin repetir	**261**	**144**	**117**

Cuadro 6: Cantidad de métricas identificadas por forma de trabajo

Definimos un total de 261 métricas de calidad. Luego de analizar estas métricas, identificamos que para 117 de ellas no es necesario que sean implementadas (a las cuales llamaremos "métricas no factibles"), ya que los problemas que pueden encontrarse con estas métricas son controlados de cierta forma por la herramienta, ya sea con cálculos automáticos que realiza o por controles de integridad referencial de la base de datos.

Este hecho es algo puntual que se debe a la herramienta PSP a partir de la cual realizamos el análisis. En el caso que se tuviera otra versión de la misma herramienta que realizara distintos controles sobre el ingreso de datos, el número de métricas factibles (son las métricas que vamos a implementar) se vería afectado. Por este motivo incluimos en nuestro análisis la definición de las métricas encontradas no factibles, ya que este estudio busca ser independiente de la herramienta con la cual se ingresan los datos, o la versión de la misma.

Sin embargo, no se implementarán las métricas no factibles en el alcance de este proyecto, ya que no se encontrarían problemas de calidad para esta instancia de la base de datos. Nuestro estudio incluye entonces la definición del total de métricas encontradas ya que se está realizando un análisis global de calidad, pero la implementación solamente de las factibles para este caso en particular.

En el Cuadro 7 se detalla la cantidad de métricas definidas por los distintos métodos (formas de trabajo), incluyendo las intersecciones entre los métodos (métricas que se encontraron por más de un método).

Forma de trabajo	Cantidad de métricas definidas	Cantidad de métricas factibles
Dimensión-factor	6	5
Exploratorio	65	24
Scripts	36	30
Exploratorio, scripts	71	22
Factor, Exploratorio	5	2
Factor, Scripts	29	13
Exploratorio, Dimensión-factor, scripts	8	7
Doc-Estimating Review Times	4	4
Grading Checklist	37	37

Cuadro 7: Cantidad de métricas identificadas por forma de trabajo

En el Cuadro 8 se muestran la cantidad de métricas de calidad definidas por factor.

Dimensión	Factor	Cantidad métricas	Cantidad métricas factibles
Completitud	Cobertura	7	7
	Densidad	47	35
Consistencia	Integridad de Dominio	41	30
	Integridad Intra-Relación	37	16
	Integridad Referencial	99	33
Exactitud	Correctitud sintáctica	11	11
	Precisión	4	2
Unicidad	Contradicción	6	4
	Duplicación	9	6

Cuadro 8: Cantidad de métricas por factor

Se diferencian las métricas que son factibles de considerar para este estudio, por dimensión-factor de calidad.

En el Cuadro 9 se muestra la cantidad de métricas por problema de calidad:

Problema de calidad	Cantidad de métricas	Cantidad de métricas factibles
Precisión incorrecta	4	2
Registro Contradictorio	6	4
Registro Duplicado	9	6
Registro Incompleto	47	35
Registro Inexistente	7	7
Reglas de Integridad de Dominio	41	30
Reglas de Integridad Intra-Relación	37	16
Reglas de Integridad Referencial	99	33
Valor Fuera de Rango	11	11

Cuadro 9: Cantidad de métricas por problema de calidad

También se diferencian las métricas que son factibles de considerar pero por problema de calidad.

4.5 Mapeo de métricas de calidad con estudio externo

Para realizar un estudio más completo de la calidad de la base de datos, se complementó la identificación de métricas de calidad realizada por nuestro estudio con las métricas encontradas en el estudio [14].
Se identifica un conjunto de métricas que no fueron identificadas por nuestro estudio. Estas métricas nos permitieron complementar las identificadas por nuestro trabajo, obteniendo como resultado una mayor cantidad y variedad de métricas posibles para PSP.

A continuación se va a detallar el proceso realizado de unión de métricas entre los dos proyectos. En la siguiente sección se procederá a explicar las razones por las cuales se obtuvieron las diferencias en cantidad y problemas de calidad entre los dos proyectos, siendo que en ambos se trata de un análisis de calidad sobre la misma base de datos.

Partiendo de la planilla generada por nosotros (ANEXO A), se procedió a la búsqueda de coincidencias en la planilla (ANEXO B *"Métricas proyecto externo (2012)"*).

En nuestra planilla (ANEXO A), se agregó una columna para mapear las métricas con las del estudio externo. En dicha columna "Id Métrica Estudio [14]" se tiene el id con el cual se corresponde en la plani-

lla (*ANEXO B*). El color verde en las celdas de esa misma columna indica que hay coincidencia, y el amarillo que no la hay.

El mapeo de métricas se realizó en 3 etapas.

1. **Búsqueda por coincidencia de factor-dimensión**: se procedió a buscar para cada factor de calidad, las coincidencias con métricas de calidad de ese mismo factor en la otra planilla. La búsqueda estaba restringida por factor.
2. **Búsqueda por coincidencia de tabla y atributos de la Base de Datos**: en esta búsqueda se intentaba encontrar coincidencias considerando la tabla y atributos de la Base de Datos a la cual hacía referencia la métrica de calidad.
3. **Búsqueda general**: finalmente se realizó una búsqueda general de coincidencias para asegurase de que se mapearan todas las métricas de calidad.

Con este método de trabajo nos aseguramos de encontrar todas las coincidencias de métricas entre los dos proyectos.

Luego para terminar de hacer la unión de métricas se procedió a marcar todas las métricas de la planilla (ANEXO B) con las cuales no encontramos coincidencias. Para esto en el propio anexo se agregaron dos columnas al formato original:

- *Add?*: Contiene los siguientes valores:
  - SI: Para indicar que se va a agregar la métrica al conjunto total.
  - NO: Para indicar que no se va a agregar.
- *Motive*: Se detalla el motivo por el cual creemos que la métrica debe ser agregada o no.

A continuación se presentan los resultados obtenidos:

- Total métricas encontradas en búsqueda de proyecto actual: 158
- Total métricas encontradas en el proyecto [14]: 107
- Métricas encontradas en común: 104
- Unión de todas las métricas: 262

En el Cuadro 10 se muestra una serie de resultados, que reflejan la cantidad de métricas coincidentes y agregadas, discriminadas por factor de calidad:

Dimensión	**Factor**	**Métricas Proyecto actual**	**Métricas en común**	**Métricas estudio [14]**	**Métricas agregadas**	**Total métricas**
Completitud	Cobertura	3	9	13	4	7 (*)
	Densidad	40	17	24	7	47
Consistencia	Int. Dominio	39	5	7	2	41
	Int. Intra-Rel	25	3	15	12	37
	Int. Referencial	76	8	31	23	99
Exactitud	Corr. Sintáctica			11	11	11
	Precisión	4	1	1		4
Unicidad	Contradicción	6				6
	Duplicación	8	1	2	1	9

Cuadro 10: Total de métricas resultante de la unión de los dos proyectos

4.5.1 Análisis de diferencias encontradas

Con los resultados de la cantidad de métricas encontradas por cada estudio a la vista, nos cuestionamos el motivo por el cual existen diferencias en las métricas encontradas entre un estudio y el otro. Esta pregunta es inevitable ya que en los dos estudios se trabajó sobre la misma temática (PSP) y sobre la misma estructura de Base de Datos.

Luego de un análisis de los resultados, identificados que los principales motivos de las diferencias se pueden deber a varios hechos puntuales, tales como:

- En el estudio [14] el análisis estaba enfocado en tiempos, defectos y tamaños de los proyectos, ya que eran los datos que se iban a utilizar para realizar ciertos análisis. Esto llevo a que no se hiciera un análisis completo, si no que se analizaran sólo los datos que se consideraban relevantes.
- En el estudio [14] no se analizaron las tablas que venían con valores predefinidos en la herramienta (ejemplo "rdefecttype"), se asumió que estaban correctas.
- Hubo un conjunto de métricas definidas en el estudio [14] de las cuales se desconocía su origen, ya que las mismas no podían ser identificadas a través de la herramienta, ni de la base de datos, ni de los scripts, ni de la grading checklist. Algunas de ellas fueron sugeridas por un experto con amplio conocimiento en PSP, y en otras se definió un rango de valores arbitrario como dominio.

Además de estos motivos puntuales, también existen diferencias en la forma particular de identificar métricas por parte de los dos estudios. Algunas de estas diferencias son las siguientes:

- **Enfoque proyecto actual:** en el análisis actual se busca evaluar la calidad de cada dato almacenado en la base de datos, y si el mismo aparecía en diferentes tablas se evaluaba más de una vez. Con esto se logra alcanzar mayor profundidad en el estudio de la calidad. Un ejemplo bien claro en donde aparecen datos conceptualmente duplicados es en los reportes al instructor (tabla *PSPAssgData*) con el resto de las tablas (ejemplo tabla *Projects*), los reportes se hacen en base a los datos de las demás tablas por ende analizamos la calidad por separado.
- **Enfoque estudio [14]:** observando sus métricas hallamos que la mayoría de las identificadas se basan en el reporte al instructor (tabla *PSPAssgData*), para poder realizar esto hay que asumir que los datos utilizados en los cálculos aplicados para el reporte son correctos. Este criterio se siguió para simplificar la implementación.

Otra diferencia identificada fue el nivel de granularidad en la que se definieron las métricas de calidad. Por ejemplo, había casos en los que una métrica de nuestro proyecto se mapeaban con varias del estudio en [14]. Por ejemplo:

- Métrica proyecto actual: "Existe al menos un registro para cada una de las fases (*PLAN, DLD, CODE, COMPILE, UT, PM*)."
- Métrica Estudio [14]:
  - *"There must be a register of the time spent on the planning phase"*
  - *"There must be a register of the time spent on the design phase"*
  - *"There must be a register of the time spent on the design review phase"*
  - *"There must be a register of the time spent on the coding phase"*
  - *"There must be a register of the time spent on the code review phase"*
  - *"There must be a register of the time spent on the unit testing phase"*
  - *"There must be a register of the time spent on the post mortem phase"*

De todas maneras estas métricas de calidad fueron mapeadas con las nuestras, por lo que se encuentran coincidencias aunque las métricas estén definidas de maneras diferentes.

Como se puede observar en los motivos anteriormente descritos, las diferencias identificadas se deben fundamentalmente a que la forma en que se llevaron a cabo los análisis no fue la misma. Esas diferencias de enfoque es lo que lleva a que haya diferencias en los resultados.

4.6 Modelo inicial de calidad de datos para PSP

Uno de los objetivos principales de este proyecto es generar un modelo inicial de calidad para el dominio bajo estudio (PSP), este modelo se define por un conjunto de dimensiones, factores, problemas y métricas de calidad. Estas características de calidad se presentan en las secciones anteriores de este capítulo, a modo de resumen se detallan en el siguiente Cuadro 11:

Dimensión	Factor	Problema de calidad	Cantidad de métricas	Cantidad de métricas factibles
Exactitud	Correctitud sintáctica	Valor fuera de rango	11	11
	Precisión	Precisión incorrecta	4	2
Completitud	Densidad	Registro incompleto	47	35
	Cobertura	Registro inexistente	7	7
Consistencia	Integridad de dominio	Reglas de integridad de dominio	41	30
	Integridad intra-relación	Reglas de integridad intra-relación	37	16
	Integridad referencial	Reglas de integridad referencial	99	33
Unicidad	Duplicación	Registro duplicado	9	6
	Contradicción	Registro contradictorio	6	4
			261	144

Cuadro 11: Resumen modelo calidad PSP

Este modelo incluye 261 métricas definidas sobre PSP, las cuales 144 son factibles que ocurran. Las restantes 117 métricas se las definen y presentan dado que el modelo de calidad es independiente de la herramienta de PSP utilizada, o sea que si en un futuro esta cambia, el conjunto de métricas (factibles o no) van a seguir siendo válidas a aplicar.

El modelo abarca cuatros dimensiones de calidad; exactitud, completitud, consistencia y unicidad. Con estos se busca obtener diferentes facetas y características de la calidad, en dónde cada una define un conjunto de factores y problemas de calidad que resaltan características específicas de la calidad.Decimos que es un modelo *inicial* de calidad de datos ya que se construye analizando los datos de una base de datos particular de PSP. La metodología presentada, métricas y problemas de calidad definidos son aplicados sobre dicha base. Para poder ser aplicados en otras bases de PSP se debería trabajar en generalizar la propuesta, en base al modelo inicial construido en este trabajo.

Además este modelo de calidad sirve como punto de partida para trabajos a futuro, como ser una limpieza de datos o una migración.

5 Medición de la calidad de los datos de PSP

En esta sección se detalla la *metadata* generada para el estudio de calidad sobre la base de datos de PSP. Se describe el modelo relacional utilizado y el significado de las tablas, así como también sus relaciones. También se explica la estrategia empleada para implementar los métodos de medición de las métricas de calidad definidas.

5.1 Modelo de metadatos de calidad

Se genera un modelo de metadatos relacional para realizar el estudio de calidad de datos sobre PSP. Este modelo busca ser lo más general posible, de manera que pueda ser reutilizado para otros estudios de calidad sobre datos en el mismo dominio de aplicación (PSP).

La metadata generada para este estudio de calidad se almacena en una nueva base de datos llamada "*calidaddedatospsp*", en caso de ya existir se genera previamente un respaldo. La misma se crea con un *script SQL* que surge de la unión de los siguientes diez scripts:

- Un *script* para crear los metadatos iniciales.
- Ocho *scripts* que corresponden a cada factor de calidad analizado.
- Un último *script* el que genera los valores de calidad.

Las tablas que definen la base de datos se pueden agrupar conceptualmente en dos niveles, como se muestra en la Figura 6.

- **Nivel 1**: el primer nivel contiene la información necesaria para representar el metamodelo de las métricas y problemas de calidad definidos, así como también la ubicación y detección de los errores encontrados en el análisis de calidad sobre las tablas de PSP. Abarca todas las tablas de la base de datos a excepción de las tres que corresponden al nivel 2.
- **Nivel 2**: en base a la información contenida en las tablas de primer nivel, se obtienen los valores de calidad por métricas, por problema de calidad y por tabla. Los valores de calidad se almacenan en las siguientes tres tablas:
    - cd_valor_de_calidad_por_metrica
    - cd_valor_de_calidad_por_prob_cal
    - cd_valor_de_calidad_por_tabla

  Para realizar los cálculos correspondientes para todos los valores de calidad, es necesario disponer de la información en las tablas de nivel 1.

El modelo se define de forma de cumplir con los siguientes tres objetivos:

1. **Ubicar los errores**: el modelo debe facilitar la ubicación del error, esto es muy importante para poder aplicar futuras limpiezas. De la manera en que fue implementada la medición y registro de resultados del análisis de calidad de datos, se podría obtener dinámicamente el registro con problema. A pesar de que las limpiezas quedan fuera del alcance del estudio de calidad, se brindan las estructuras necesarias para facilitar su ejecución en un trabajo futuro.
2. **Conocer las causas de los errores**: para cada error detectado, es deseable conocer la posible causa del mismo. Es por esto que al ejecutar cada consulta SQL, se guarda información que ayudará a conocer sus posibles causas de ocurrencia. Esto contribuye también en proponer acciones de prevención asociadas, de forma de evitar que el error vuelva a suceder en futuras instancias.
3. **Obtener valores de calidad**: una vez ejecutadas todas las métricas, el modelo debe permitir obtener valores de calidad para conocer la calidad de la base de datos estudiada. En la sección 6 "*Resultados obtenidos*" se definen estos valores y sus cálculos. Esto nos permite además poder ejecutar el análisis de calidad sobre los mismos datos pero en diferentes momentos (por ejemplo: sobre la base de datos tal cual está, o luego de aplicar una limpieza de datos), para luego comparar los resultados y analizar el impacto.

En la Figura 6 se muestra el esquema de base de datos "*calidaddedatospsp*", indicando sus respectivos niveles.

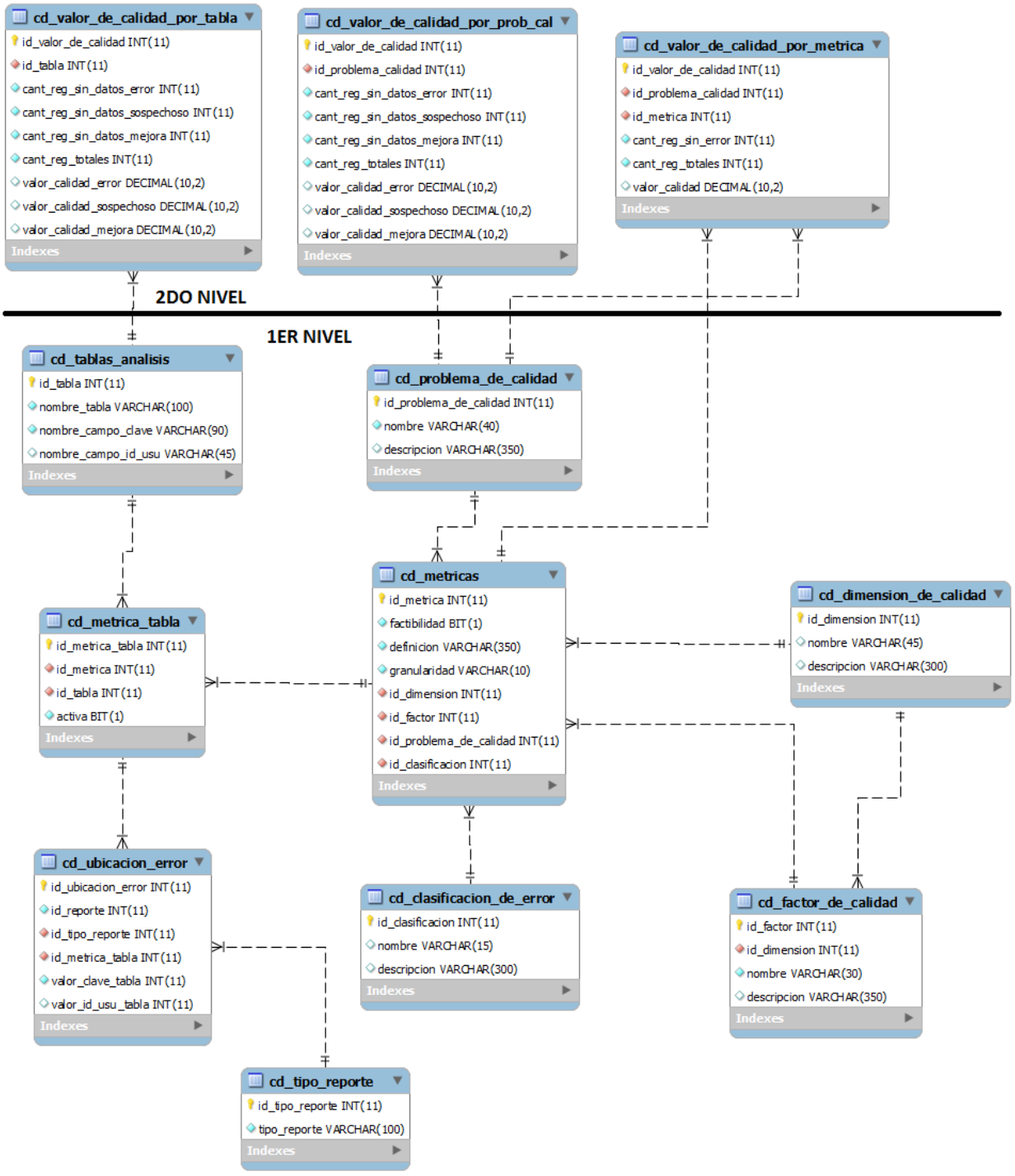

Figura 6: Estructura del modelo de metadatos

La tabla *cd_ubicacion_error* es la que contiene la información sobre donde se encuentran almacenados todos los errores detectados en la metadata. Esta tabla se relaciona con *cd_tipo_reporte*, que sirve para agrupar los errores por factor de calidad, por lo tanto en esta tabla hay un registro por cada factor de calidad.

En la tabla *cd_ubicacion_error* se encuentran los identificadores (ids) a las tablas de reportes. Estas tablas (detalladas en el ANEXO H *"Complemento esquema base de datos"*) contienen la información completa de los errores detectados. Hay una tabla de reportes por cada factor de calidad. Los nombres de las mismas comienzan con el prefijo "cd_reporte_", y almacenan la información completa sobre el error (tabla, campos en los que se presentan, valores de los campos). Por lo tanto, la tabla *cd_ubicacion_error* solo contiene los ids a las filas de las tablas de reportes, de manera de centralizar todos los errores en un solo lugar.

A su vez la tabla *cd_ubicacion_error* se relaciona con la tabla *cd_metrica_tabla*, que contiene las métricas y las tablas relacionadas con dichas métricas, ya que cada métrica puede tener una o dos tablas asociadas. La tabla *cd_metrica_tabla* contiene una Foreing Key (de aquí en más FK) a la tabla que contiene las tablas de la base de datos (*cd_tabla_analisis*), y una FK a la tabla que contiene las métricas (*cd_metricas*).

Las métricas están clasificadas de acuerdo a la dimensión, factor de calidad y problema de calidad. Es por eso que la tabla de las métricas se relaciona con las tablas *cd_dimension_de_calidad*, *cd_factor_de_calidad* y *cd_problema_de_calidad*.

Finalmente, cada métrica está clasificada de acuerdo a si se trata de un error, de un registro sospechoso no necesariamente error, o si es una oportunidad de mejora. Por esto la tabla *cd_metrica* también está relacionada con la tabla *cd_clasificacion_error* que es la que contiene las diferentes clasificaciones.

A continuación se describe la función que cumple cada una de las tablas en el estudio de calidad.
Tablas del primer nivel:

- **cd_problema_de_calidad:** contiene todos los problemas de calidad analizados en este estudio.
  Consta de los siguientes campos:
  - *id_problema_de_calidad*: identificador de la tabla. Primary Key (de aquí en más PK).
  - *nombre*: nombre del problema de calidad.
  - *descripcion*: descripción complementaria.
- **cd_dimension_de_calidad:** contiene las dimensiones de calidad analizadas.
  Consta de los siguientes campos:
  - *id_dimension*: identificador de la tabla (PK).
  - *nombre*: nombre de la dimensión de calidad.
  - *descripcion*: descripción complementaria.
- **cd_factor_de_calidad:** contiene los factores de calidad analizados.
  Consta de los siguientes campos:
  - *id_factor*: identificador de la tabla (PK).
  - *nombre*: nombre del factor de calidad.
  - *descripcion*: descripción complementaria.
- **cd_clasificacion_de_error:** contiene las clasificaciones de los errores.
  Consta de los siguientes campos:
  - *id_clasificacion*: identificador de la tabla (PK).
  - *nombre*: nombre de la clasificación.
  - *descripcion*: descripción complementaria.
- **cd_metricas:** contiene todas las métricas encontradas durante la exploración y búsqueda de posibles problemas en la base de datos de PSP. Contiene la información presentada en la planilla del ANEXO A.
  Consta de los siguientes campos:
  - *id_metrica*: identificador de la tabla (coincide con el número de métrica en la planilla).
  - *factibilidad*: indica si la métrica es factible de aplicar de acuerdo al contexto.
  - *definicion*: descripción de la métrica.

  - *granularidad*: indica si la métrica se aplica a nivel de Celda, Tupla o Tabla.
  - *id_dimension*: FK a la tabla cd_dimension_de_calidad.
  - *id_factor*: FK a la tabla cd_factor_de_calidad.
  - *id_clasificacion*: FK a la tabla cd_clasifiacion_de_error.
  - *id_problema_de_calidad*: FK a la tabla cd_problema_de_calidad.
- **cd_tablas_analisis:** contiene los nombres de todas las tablas de la base de datos de PSP. Esto permite que, a la hora de realizar el estudio de calidad, las tablas pueden ser referenciadas mediante identificadores (y no mediante nombres).
  Consta de los siguientes campos:
  - *id_tabla*: identificador de la tabla.
  - *nombre_tabla*: nombre de la tabla en la base de datos.
  - *nombre_campo_clave*: nombre de la primary key de la tabla.
  - *nombre_campo_id_usu*: id del usuario que ingreso los datos a través de los formularios de PSP.
- **cd_metrica_tabla:** contiene la relación métrica-tabla (para una métrica, todas las tablas que involucra).
  Consta de los siguientes campos:
  - *id_metrica_tabla*: identificador de la tabla
  - *id_metrica*: FK a la tabla cd_metricas
  - *id_tabla*: FK a la tabla cd_tablas_analisis
- **cd_tipo_reporte:** contiene los tipos de reportes, se genera uno por cada factor (más adelante se detalla el propósito de los reportes).
  - *id_tipo_reporte*: identificador de la tabla.
  - *tipo_reporte*: nombre del factor de calidad.
- **cd_ubicacion_error:** contiene la ubicación de cada error detectado por la ejecución de las métricas.
  Consta de los siguientes campos:
  - *id_ubicacion_error*: identificador de la tabla
  - *id_reporte*: es el id correspondiente a la tabla cd_reporte_x, en la cual se encuentra la tupla con error. Este campo si bien tiene un id no es una FK, ya que es un identificador a diferentes tablas. El campo *id_tipo_reporte* es el que diferencia a qué tabla con prefijo "cd_reporte" se hace referencia.
  - *id_tipo_reporte*: es una FK a la tabla cd_tipo_reporte. Con esto sabemos a qué reporte corresponde cada error.
  - *id_metrica_tabla*: es una FK a la tabla cd_metrica_tabla.
  - *valor_clave_tabla*: este registro indica el valor de la PK de la tabla de la base de datos en la cual efectivamente se encuentra el error. Ya que mediante la FK anterior tenemos acceso a la tabla en que se encuentra el error, con este campo tenemos además la tupla exacta de la tabla.
  - *valor_id_usu_tabla*: finalmente se tiene el identificador del usuario, ya que en las tablas de la base hay registros con los mismos identificadores pero con diferente usuario.

Tablas del segundo nivel:

- **cd_valor_de_calidad_por_metrica:** contiene los valores de calidad asociados a cada métrica definida.
  Consta de los siguientes campos:
  - *id_valor_de_calidad*: identificador de la tabla.
  - *id_problema_de_calidad*: FK a la tabla cd_problema_de_calidad.

- *id_metrica*: FK a la tabla cd_metricas
- *cant_reg_sin_error*: en este campo se almacenan la cantidad de registros sin error para la tabla asociada a la métrica dada por id_metrica.
- *cant_registros_totales*: cantidad de registros totales de la tabla asociada a la métrica dada por id_metrica.
- *valor_calidad*: es el resultado dado por la cuenta: (cantidad de registros sin error)/(cantidad de registros totales).

- **cd_valor_de_calidad_por_prob_cal:** contiene los valores de calidad asociados a cada problema de calidad definido.
  Consta de los siguientes campos:
  - *id_valor_de_calidad*: identificador de la tabla.
  - *id_problema_de_calidad*: FK a la tabla cd_problema_de_calidad.
  - *cant_reg_sin_datos_error*: en este campo se almacenan la suma de las cantidades de registros sin error de todas las tablas asociadas al problema de calidad id_problema_de_calidad.
  - *cant_reg_sin_datos_sospechoso*: en este campo se almacenan la suma de las cantidades de registros sin datos sospechosos, de todas las tablas asociadas al problema de calidad id_problema_de_calidad.
  - *cant_reg_sin_datos_mejora*: en este campo se almacenan la suma de las cantidades de registros sin datos con oportunidad de mejora, de todas las tablas asociadas al problema de calidad id_problema_de_calidad.
  - *cant_registros_totales*: corresponde a la suma de todos los registros de todas las tablas asociadas al problema de calidad id_problema_de_calidad.
  - *valor_calidad_error*: es el valor de calidad dado por la cuenta: (*cant_reg_sin_datos_error*)/(*cant_registros_totales*).
  - *valor_calidad_sospechoso*: es el valor de calidad dado por la cuenta: (*cant_reg_sin_datos_sospechoso*)/(*cant_registros_totales*).
  - *valor_calidad_mejora*: es el valor de calidad dado por la cuenta: (*cant_reg_sin_datos_mejora*)/(*cant_registros_totales*).
- **cd_valor_de_calidad_por_tabla:** contiene los valores de calidad asociados a cada tabla de la base de datos de PSP.
  Consta de los siguientes campos:
  - *id_valor_de_calidad*: identificador de la tabla.
  - *id_tabla*: FK a la tabla cd_tablas_analisis.
  - *cant_reg_sin_datos_error*: se almacenan la cantidad de registros sin error para la tabla dada por id_tabla.
  - *cant_reg_sin_datos_sospechoso*: se almacenan la cantidad de registros sin datos sospechosos para la tabla dada por id_tabla.
  - *cant_reg_sin_datos_mejora*: se almacenan la cantidad de registros sin datos con oportunidad de mejora para la tabla dada por id_tabla.
  - *cant_reg_totales*: total de registros de la tabla dada por id_tabla.
  - *valor_calidad_error*: es el valor de calidad dado por la cuenta: (*cant_reg_sin_datos_error*)/(*cant_registros_totales*).
  - *valor_calidad_sospechoso*: es el valor de calidad dado por la cuenta: (*cant_reg_sin_datos_sospechoso*)/(*cant_registros_totales*).
  - *valor_calidad_mejora*: es el valor de calidad dado por la cuenta: (*cant_reg_sin_datos_mejora*)/(*cant_registros_totales*).

El esquema consta además de otras tablas que por motivos de espacio no se presentan en este informe. Por ejemplo, las tablas con prefijo “cd_reporte”, que son las tablas que contienen la descripción y ubicación del error, y las tablas auxiliares con prefijo “cd_aux” y “cd_temp” que son utilizadas temporalmente como ayuda a la hora de la ejecución de los scripts. Estas tablas y su descripción se pueden ver en el ANEXO H.

5.2 Implementación de los métodos de medición

En esta sección se detalla el método seleccionado para implementar los métodos de medición de las métricas categorizadas como factibles. Además se define la estrategia aplicada para lograr una implementación ordenada y mantenible.

5.2.1 Método de medición

El método de medición empleado son consultas SQL, esto es, para cada métrica factible se realiza una consulta SQL que la implementa. Dada la importancia del resultado de la medición, es fundamental conocer cuál es y dónde se encuentra cada posible error identificado. Con el modelo de datos y metadatos mencionado en la sección 5.1 *“Modelo de metadatos de calidad”*, se puede conocer la tabla y registro que contiene un error, así como también información complementaria para ayudar a comprender la posible causa de su ocurrencia.

5.2.2 Estrategia desarrollada

Dado el volumen de métricas encontradas y la complejidad de cada una, se decide agrupar su implementación según el factor de calidad que corresponda. Es por eso que se crea un script SQL por cada factor de calidad analizado. Los ochos script originales fueron evolucionando hasta llegar a la solución definitiva; una vez encontrada esta, se unieron con el script inicial de generación de metadatos y el de generación de valores de calidad para formar un único script centralizado. Se elige esta estrategia ya que las métricas de un mismo factor poseen características similares, facilitando su implementación.
Estos scripts se encuentran detallados en el ANEXO F *“Scripts SQL de generación de metadata del análisis de calidad”*.

6 Resultados obtenidos

Uno de los resultados más importantes que se desprenden de este estudio es la obtención de los valores de calidad. Estos ayudan a conocer el estado de la base de datos analizada, además permiten identificar dónde se concentran la mayor cantidad de errores de calidad y a qué se deben.

A partir del análisis de los valores de calidad es posible obtener cierta información relevante. Por ejemplo, se podría conocer si los errores se deben a que la base de datos analizada está mal modelada, si corresponden al mal uso de la herramienta, o la misma no contiene las validaciones suficientes, problemas de migración, entre otras posibles causas.

Se presentan tres tipos de valores diferentes, de acuerdo a los tres grandes conceptos de estudio definidos: valores de calidad por métrica, valores de calidad por problema de calidad y valores de calidad por tabla.

Todos los valores de calidad se calculan de la siguiente forma:

(cantidad de tuplas sin error) / (cantidad de tuplas totales)

De aquí en más llamaremos a este cálculo "porcentaje de tuplas correctas", donde las tuplas equivalen a los registros de las tablas.

En las próximas subsecciones se explica cómo se calcula el valor de calidad para cada uno de los tres tipos definidos.

Los datos que se muestran en las siguientes secciones son el resultado de la ejecución del script "*10-script_generacion_valores_calidad.sql*" (detallado en el ANEXO F) sobre las tablas de primer nivel. Este script calcula todos los valores de calidad y los almacena en las tablas de segundo nivel anteriormente descriptas (en la sección 5.1).

A su vez, se realiza un análisis sobre los menores valores de calidad, presentando un estudio de las posibles causas de esos valores.

Del análisis del conjunto de valores de calidad hallados se desprenden diversas conclusiones, ya que estos son un punto de partida para encontrar defectos en el diseño de la base de datos de PSP, así como errores de la herramienta, o detectar un mal uso de la misma.

Adicionalmente a estos estudios, en base a la *metadata* generada, se obtienen algunos datos de interés que se mencionan en la última sección (6.5 "*Análisis transversal"*).

6.1 Consideraciones sobre las métricas de calidad excluidas

Previo a la ejecución de cualquier análisis de calidad de datos, es necesario considerar cuáles son los datos y métricas que resultan relevantes para el dominio de aplicación particular que se está analizando.
Por este motivo, para el análisis de los valores de calidad que se obtienen en este estudio, se excluyeron algunas métricas que se encuentran definidas en la planilla detallada en el ANEXO A.

En el primer análisis que se realizó se incluyeron todas las métricas factibles encontradas en el anexo mencionado anteriormente. En este análisis se detectó que un subconjunto de esas métricas tenía un alto impacto en los valores de calidad obtenidos por problema de calidad y por tablas, alcanzando valores muy bajos.

Cuando se siguió profundizando en el análisis con el fin de encontrar la causa de estos bajos valores, se encontró que el problema se originaba en el *script* utilizado para la migración de datos hacia la base de datos de estudio. En dicho *script*, donde se cargan todos los datos en cada una de las tablas de la base de estudio, se observó que en algunos casos se insertaron valores que no son correctos. Los valores incorrectos encontrados con mayor frecuencia fueron el valor de tipo texto *"Hay Texto"*, o para columnas que contiene fechas, el valor que se encontró muy a menudo fue *"1000-01-01 00:00:00"*.

A continuación se muestra a modo de ejemplo, el análisis del valor de calidad obtenido para una de las métricas excluidas del análisis:

idMetrica	56
Descripción	Cuando se finaliza un programa es guardado como una parte. El nombre de dicha parte se debe actualizar en caso de que se cambie el nombre del programa finalizado
Valor	0.08
Causa	La principal causa que justifica el bajo valor de calidad para esta métrica, proviene de analizar los datos de la tabla "rparts", que es donde se almacenan las partes ingresadas por los usuarios. En dicha tabla se puede apreciar, que casi la totalidad de los registros, en el campo "RContainerName" contienen el valor "Hay Texto". Se detectó que estos datos erróneos no son procedentes de la herramienta "*PSP Student Workbook*", sino que se debe a problemas generados en el *script* de migración mencionado anteriormente. Es ahí donde se inyectan los registros erróneos, con el valor "Hay Texto".

Conocido este error, se consultó con los responsables de los datos quienes indicaron que estos valores incorrectos fueron ingresados de esa forma, ya que en el momento de la migración de las bases no eran considerados relevantes. Por dicho motivo, y para que estos problemas particulares no afecten los resultados obtenidos en este proyecto, se optó por no considerar las métricas que acarrean dichos problemas a la hora de realizar el cálculo de los valores de calidad, tanto por métricas, como por problemas de calidad y por tablas.

El conjunto de métricas que fueron consideradas y excluidas para el cálculo de los valores de calidad, se pueden apreciar en el ANEXO A, diferenciadas por la columna titulada: "Definida sobre datos con errores de migración".

6.2 Valores de calidad por métrica

Para hallar los valores de calidad por métrica, se realiza el porcentaje de tuplas correctas. Se obtiene la cantidad de registros por métrica existentes en la tabla "cd_ubicacion_error", la cual corresponde a la cantidad de errores detectados en la tabla implicada a la métrica. Luego se calcula la cantidad de registros totales de esta tabla, para finalmente calcular el porcentaje:

(Total de tuplas totales - Total de tuplas con error) / (Total de tuplas totales)

En la Figura 7 se muestra un esquema de las tablas principales de la metadata con valores genéricos y como se relacionan entre sí. En este esquema, la tabla "cd_ubicacion_error" tiene un rol central, ya que es a través de ella que se puede acceder a todos los errores encontrados (como fue descripto en la sección 5.1).

En la tabla "cd_ubicacion_error" se encuentran los registros de todas las ubicaciones de los errores encontrados. Esta tabla contiene el identificador (id_reporte) que corresponde al identificador de un registro en algunas de las tablas con prefijo "cd_reporte" (dependerá a cuál reporte corresponda el error, existe uno por cada factor de calidad), que son las que contienen la información de cuál es el error. Para diferenciar la tabla con prefijo "cd_reporte" a la que se está haciendo referencia, se tiene la columna llamada "id_tipo_reporte". Ya que como se tienen los id de todas las tablas con prefijo "cd_reporte" (son ocho), estos pueden coincidir, pero al tener el par (id_reporte, id_tipo_reporte) cada fila se identifica sin ambigüedades. La tabla "cd_ubicacion_error" también cuenta con una *foreing key* a la tabla "cd_metrica_tabla". Esta última lo que hace es indicar qué métrica y qué tabla están asociadas al error.

Por lo tanto, la tabla "cd_ubicacion_error" contiene una referencia al registro con error así como también una referencia a cuál es el error (referencia a una tabla con prefijo "cd_reporte").

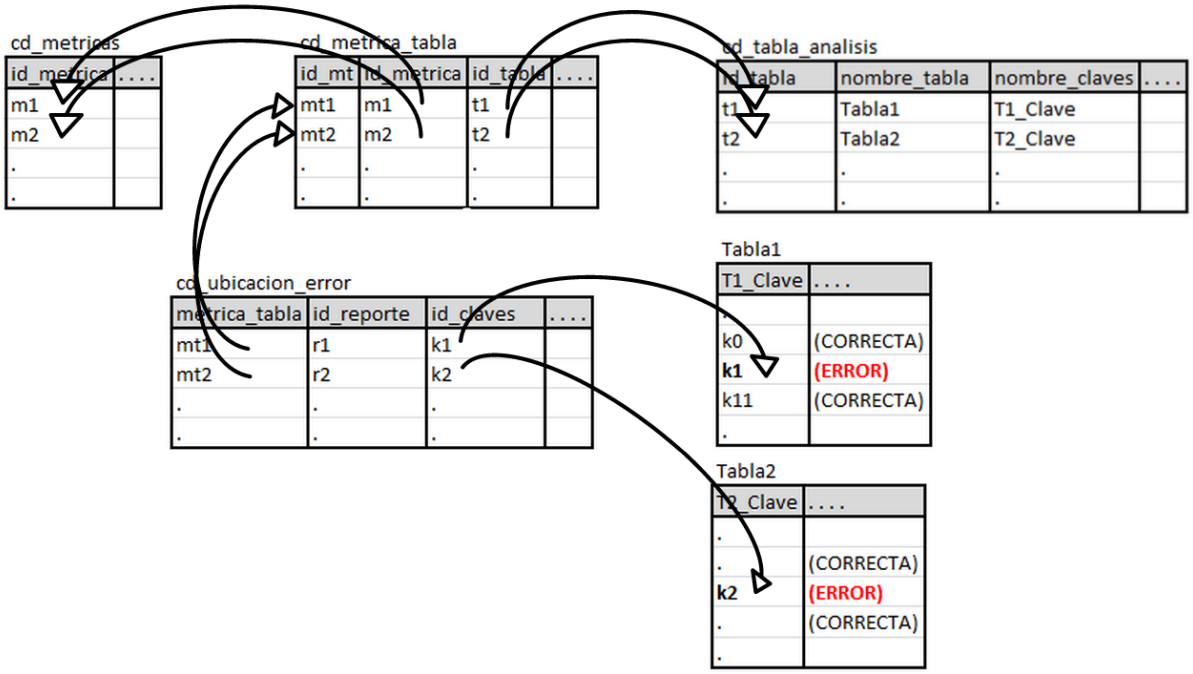

Figura 7: Ejemplo genérico de la tabla “cd_ubicacion_error”

Se escogió un caso real para presentar una instancia del esquema de la Figura 7. Este ejemplo se ilustra en la Figura 8, se escogieron las siguientes métricas:

- **Métrica 18:** El tiempo en corregir un defecto debe ser mayor que 0 (Reglas de integridad de dominio)
- **Métrica 236:** Los formularios PIP, deben contener una descripción de mejora (Registro Incompleto)

Estas métricas fueron seleccionadas debido a que ambas influyen en que los valores de calidad obtenidos para los problemas de calidad mencionados anteriormente no sean los óptimos.

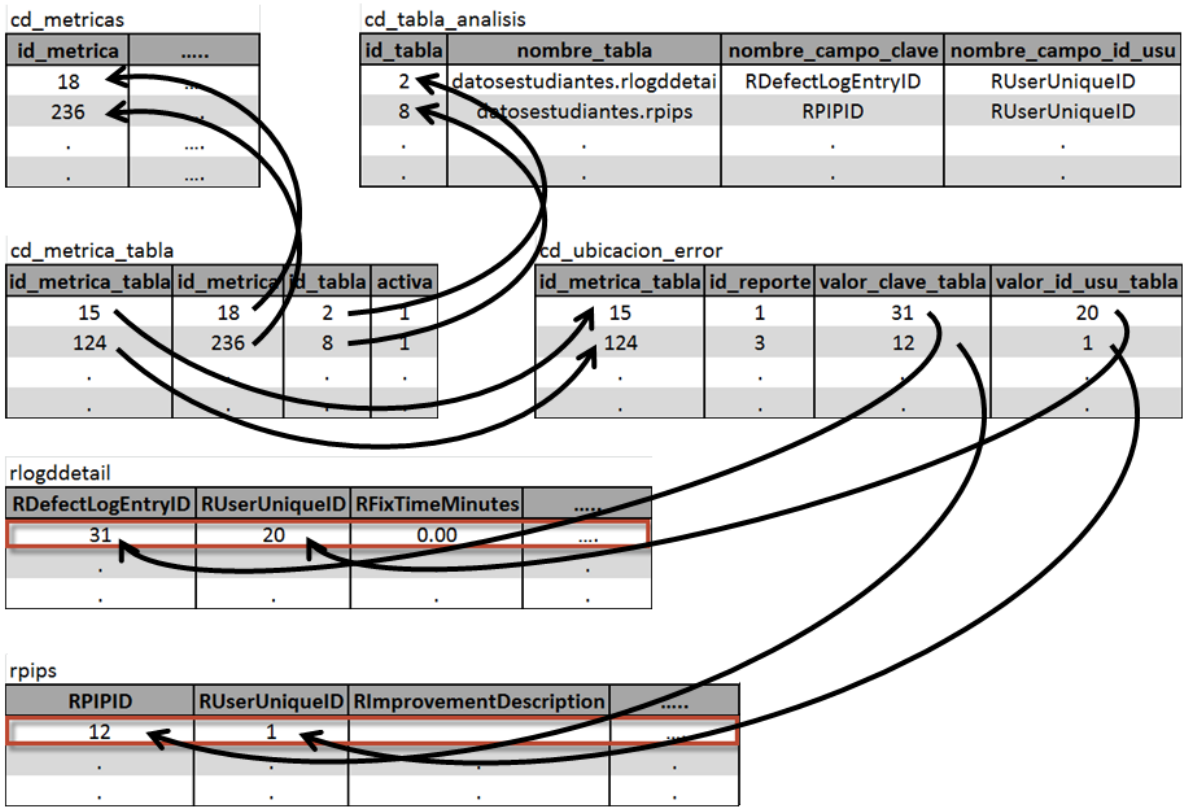

Figura 8: Ejemplo ilustrativo de la tabla "cd_ubicacion_error

Se puede apreciar en la Figura 8, que la tabla que concentra la información utilizada para identificar los registros problemáticos es la tabla "cd_ubicacion_error". Por medio de la columna "id_metrica_tabla", se puede navegar hasta llegar al detalle de la métrica correspondiente al registro analizado, como así también a la tabla asociada a dicha métrica, y los valores claves de la misma ("nombre_campo_clave" y "nombre_campo_id_usu"). A partir de estos valores, si se combinan con las columnas ("valor_clave_tabla" y "valor_id_usu_tabla") de la tabla "cd_ubicacion_error", se logra identificar precisamente cual es la túpla dentro de cada tabla asociada al posible error, mejora o registro sospechoso registrado en el esquema.

Mediante la identificación de cada uno de los errores, es que se realizan los posteriores cálculos de los valores de calidad que se mostrarán a continuación.

Luego de ejecutar el script SQL (mencionado en la sección 5.1 *"Modelo de metadatos de calidad")*, se almacenan todos los valores de calidad asociados a cada métrica en la tabla "cd_valor_de_calidad_por_metrica".

Los valores de calidad calculados se pueden apreciar en el ANEXO D *"Valores de calidad por métrica"*.

Este cálculo implementado en el script SQL, lo podemos traducir al siguiente pseudocódigo:

```
listadoProblemasCal = ObtenerTodosProblemasCalidad();
Para cada probCal en listadoProblemasCal
{
        Listado(metrica,tabla) = ObtenerTodasMetricasTabla(probCal);
        Para cada par (metrica,tabla) en Listado
```

```
        {
        cantRegConError =  ObtenerCantRegConError(probCal,metrica,tabla);
            cantTotalReg =  ObtenerCantTotalReg(tabla);
            cantRegSinError = cantTotalReg - cantRegConError;
            valorCalidad = cantRegSinError / cantTotalReg;
           GuardarValorCalidadEnBD(tabla, métrica, probCal, cantRegSinError,
               cantTotalReg, valorCalidad);
        }
}
```

Cuadro 12: Cálculo de valores de calidad por métrica

A continuación se detallan todos los métodos presentados en el pseudocódigo del Cuadro 12:

- *ObtenerTodosProblemasCalidad()*: obtiene todos los problemas de calidad definidos y analizados.
- *ObtenerTodasMetricasTabla(probCal)*: se obtienen todas las métricas asociadas al problema de calidad y su tabla activa asociada (recordar que en este estudio cada métrica puede estar asociada a más de una tabla pero solo una es elegida como activa, que es la que contiene los errores).
- *ObtenerCantRegConError(probCal,metrica,tabla)*: obtiene todos los registros con error asociados a la tabla, a la métrica y al problema de calidad pasados como parámetros.
- *ObtenerCantTotalReg(tabla)*: obtiene la cantidad de registros que posee la tabla "*tabla*" pasada como parámetro.
- *GuardarValorCalidadEnBD(tabla,métrica,probCal,cantRegSinError,cantTotalReg,valorCalidad)*: agrega un registro en la tabla *cd_valor_de_calidad_por_metrica* con la información pasada como parámetro:
    - *tabla*: tabla analizada.
    - *métrica*: métrica que involucra a la tabla analizada.
    - *probCal*: problema de calidad analizado.
    - *cantRegSinError*: cantidad de registros sin error de la tabla "*tabla*" asociada a la métrica y al problema de calidad "*probCal*".
    - *cantTotalReg*: corresponde a la suma de total de registros de la tabla asociada a la métrica "*métrica*".
    - *valorCalidad*: valor de calidad hallado para la métrica "*métrica*".

Las métricas se dividen en métricas que clasifican errores, datos sospechosos o mejoras:

- **Error**: corresponde a registros donde existe la certeza que contienen un error. Por ejemplo, si existe un registro donde la cantidad de ítems reales de las partes adicionales es 0, tenemos la certeza de que ese registro no contiene un dato válido.
- **Sospechoso**: corresponde a registros que son candidatos a ser errores pero que no existe la seguridad para afirmarlo. Por ejemplo, si un usuario supera la tasa de remoción de 5 defectos por hora para las revisiones de diseño, es un dato que no cumple con esa condición recomendada por PSP pero no es necesariamente considerado un error.
- **Mejora**: son registros en los cuales surgen oportunidad de mejora. Por ejemplo, fechas que son consideradas inconsistentes porque no contiene la precisión de las horas, minutos y segundos, se identifica como una mejora que consiste en ajustar la precisión de la fecha.

En la Figura 9 se presentan la cantidad de métricas por clasificación, distribuidas según el porcentaje de datos correctos.

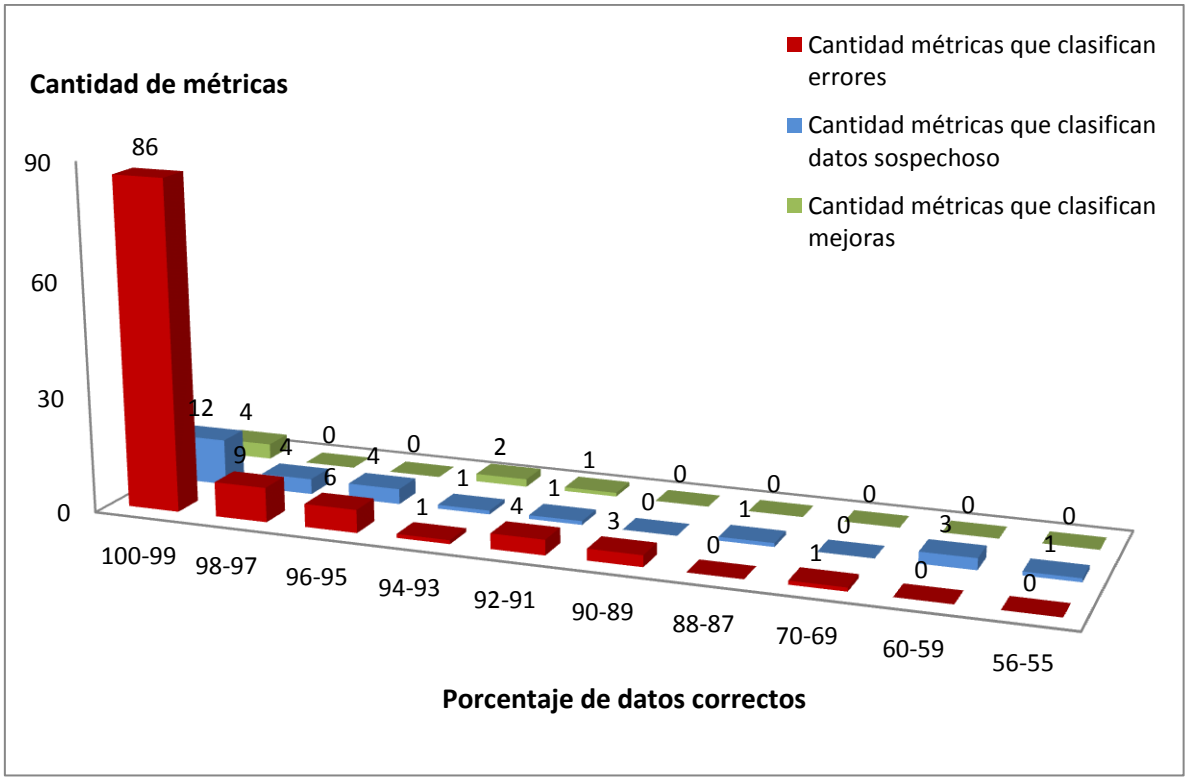

Figura 9 - Distribución cantidad de métricas por % de datos correctos

Dada la cantidad de métricas implementadas, se seleccionan algunas representativas y con un valor de calidad (para este cálculo no se consideran las clasificaciones de las métricas) inferior al 0.90, o sea que tienen hasta el 90% de tuplas correctas.

En el Cuadro 13 se presentan las ocho métricas que tienen un valor de calidad inferior a 0.90, en la columna "Clasificación Error" como fue clasificada.

Id Métrica	Cantidad de Registros Sin Error	Clasificación Error	Cantidad de Registros Totales	Valor de Calidad
225	2028	Sospechoso	3648	0.56
227	2166	Sospechoso	3648	0.59
228	2151	Sospechoso	3648	0.59
226	2175	Sospechoso	3648	0.60
59	3215	Mejora	4565	0.70
223	3164	Sospechoso	3648	0.87
181	40621	Error	45419	0.89
180	3072	Error	3465	0.89

Cuadro 13: Las ocho métricas con valor de calidad inferior a 0.90

Para los valores de calidad analizados, se buscan todas las posibles causas analizando tanto la herramienta, como la base de datos y el script de carga de datos.

De este conjunto de ocho métricas, las que poseen los identificadores 223, 225, 226, 227, 228, corresponden a métricas que miden el cumplimiento de criterios recomendados por PSP. Por ser un

criterio deseable y no algo estrictamente restrictivo, los casos detectados refieren al incumplimiento de estos criterios pero no necesariamente a un registro con error.

A continuación se detallarán las causas de los bajos valores de calidad de las ocho métricas antes mencionadas.

idMetrica	225
Descripción	La tasa de revisión de diseño estimado no puede exceder de 200 LOCS por hora.
Valor	0,56
Causa	Se puede observar que estos bajos valores de calidad, se deben a que gran cantidad de los usuarios estiman sus revisiones de diseño a niveles superiores a los 200 LOCS por hora. Los valores están bien registrados por la herramienta, el problema se debe a que los usuarios le dedican poco esfuerzo y tiempo en esta fase de revisión en relación al tamaño total del programa, esto hace que sus estimaciones para la revisión de diseño sean sobrestimadas.

idMetrica	227
Descripción	La tasa de revisión de diseño real no puede exceder de 200 LOCS por hora.
Valor	0,59
Causa	Se puede observar que estos bajos valores de calidad, se deben a que gran cantidad de los usuarios realiza sus revisiones de diseño a niveles superiores a los 200 LOCS por hora. Los valores están bien registrados por la herramienta, el problema se debe a que los usuarios le dedican poco tiempo y esfuerzo en esta fase de revisión en comparación con el tiempo total del programa. Esto lleva a la interrogante y posible motivo de análisis (que excede el alcance de este proyecto), de si los usuarios lograron realmente realizar sus revisiones de diseño a un ritmo superior al sugerido, o si esta taza de 200 LOCS por hora, debería ser analizada y reajustada.

idMetrica	228
Descripción	La tasa de revisión de código real no puede exceder de 200 LOCS por hora.
Valor	0,59
Causa	Se puede observar que estos bajos valores de calidad, se deben a que gran cantidad de los usuarios realiza sus revisiones de código a niveles superiores a los 200 LOCS por hora. Los valores están bien registrados por la herramienta, el problema se debe a que los usuarios le dedican poco esfuerzo y tiempo en esta fase de revisión en al tamaño total del programa. Esto lleva a la interrogante y posible motivo de análisis (que excede el alcance de este proyecto), de si los usuarios lograron realmente realizar sus revisiones de código a un ritmo superior al sugerido, o si esta taza de 200 LOCS por hora, debería ser analizada y reajustada.

idMetrica	226
Descripción	La tasa de revisión de código estimado no puede exceder de 200 LOCS por hora.
Valor	0,6
Causa	Se puede observar que estos bajos valores de calidad, se deben a que gran cantidad de los usuarios estima sus revisiones de código a niveles superiores a los 200 LOCS por hora. Los valores están bien registrados por la herramienta, el problema se debe a que los usuarios le dedican poco tiempo y esfuerzo en esta fase de revisión en relación al tamaño total del programa, esto hace que sus estimaciones para la revisión de código sean sobrestimadas.

idMetrica	59
Descripción	La fecha de registro PIP debe ser mayor que la fecha de inicio del programa.
Valor	0,7
Causa	Se observa que el 4% de los errores detectados se debe a que los registros PIP tienen la fecha con el valor "1000-01-01 00:00:00", como ya se mencionó este valor proviene directamente del script de migración. Para este caso no se excluyó la métrica dado que esta inconsistencia generada en la migración tiene un impacto mínimo en el valor de calidad final. Los demás errores, o sea los 96% restantes, surgen debido que al registrar la fecha del registro PIP, la misma se guarda con el valor 0 para las horas, minutos y segundos (00:00:00). Este problema se genera por el mal uso de los calendarios utilizados por la herramienta para los registros de fechas.

idMetrica	223
Descripción	No debe haber más de 5 defectos por hora en la fase de revisión de diseño.
Valor	0,87
Causa	Como indica esta métrica, la tasa de remoción de defectos no debe superar los 5 por hora para las revisiones de diseño. Analizando los datos generados y la herramienta, se concluye que los usuarios detectan y remueven muchos defectos en la revisión de diseño en relación al tiempo dedicado en esta fase. De todo este conjunto de errores el rango de defectos removidos oscila entre 1 a 19 defectos y en un tiempo entre 1 a 148 minutos.

idMetrica	181
Descripción	Dada una fase *i*, la fecha de inicio de dicha fase debe ser mayor o igual a la fecha de fin de todas las fases que le anteceden en secuencia a la fase *i*.
Valor	0,89
Causa	Al iniciar una nueva fase, ni la herramienta ni las reglas definidas sobre la base de datos de PSP validan que se hayan finalizado las fases que le preceden. Por esta razón, muchas veces los usuarios que utilizan la herramienta, registran el inicio de una nueva fase olvidando finalizar la anterior. Además la herramienta permite registrar fases que preceden a las que ya existen, esto implica que la fecha de finalización de dicha fase sea mayor que las de inicio de las fases posteriores ya registradas. Por ejemplo si se registra la fase de codificación y ya existe la fase de compilación registrada, las fechas de estas fases no serán consistentes.

idMetrica	180
Descripción	La fecha de creación de un defecto inyectado en una fase, debe ser mayor o igual a la fecha de comienzo del proyecto.
Valor	0,89
Causa	Analizando este valor de calidad, se detecta que el 11% de los programas tienen su fecha de inicio inconsistente con algunas de las fechas de registro de sus defectos. Se comparan todas las fechas que generan estas diferencias, y se detectó lo siguiente:

	1. El 60% de estos errores se debe al mal registros del tiempo para el cual fue detectado el defecto. Por ejemplo, si al programa lo iniciamos el día "dd/mm/yyyy hh:mm:ss", y ese mismo día registramos un defecto, el mismo será almacenado de la forma "dd/mm/yyyy 00:00:00", esto es un bug que posee el calendario de la herramienta. 2. El resto de las fechas están con la granularidad correcta pero son inconsistentes, esto seguramente se deba a una distracción o confusión al momento de registrar las fechas de los defectos, al elegir una fecha anterior a la actual, si no se presta atención se puede seleccionar la fecha incorrecta y el sistema no validará su correctitud.

Cuadro 14: Causas de los valores de calidad de las ocho métricas

6.3 Valores de calidad por problema de calidad

Para poder obtener los valores de calidad se debe tener en cuenta las tres posibles clasificaciones de los registros encontrados por las métricas (error, sospechoso y mejora). Esto se debe a que no se pude calificar de igual manera un registro con error de uno con un valor sospechoso, lo mismo con los datos que se detectan para mejoras. Por esta razón se obtienen tres tipos de valores de calidad, uno por cada clasificación.

Estos valores se calculan como el porcentaje de las tuplas correctas asociadas a todas las tablas correspondientes al problema de calidad. Esto implica sumar los "errores" (error, sospechoso o mejora) encontrados en cada tabla involucrada al problema de calidad y dividirlo entre la suma total de registros de todas las tablas asociadas al mismo. Este cálculo implementado en el script SQL se describe como pseudocódigo para que se pueda entender mejor el cálculo:

```
listadoProblemasCal = ObtenerTodosProblemasCalidad();
Para cada probCal en listadoProblemasCal
{
    listadoTablas = ObtenerTodasTablas(probCal);
    cantRegConError = 0;
    cantRegSospechosos = 0;
    cantRegConMejora = 0;
    cantTotalReg = 0;
    Para cada tabla en listadoTablas
        {
          cantRegConError = cantRegConError + ObtenerCantRegConError(probCal, tabla);
          cantRegSospechosos = cantRegSospechosos + ObtenerCantRegSospechosos (probCal, tabla);
          cantRegConMejora = cantRegConMejora + ObtenerCantRegConMejora(probCal, tabla);
          cantTotalReg = cantTotalReg + ObtenerCantTotalReg(tabla);
    }
    cantRegSinError = cantTotalReg - cantRegConError;
    cantRegSinRegSospechosos = cantTotalReg - cantRegSospechosos;
    cantRegSinRegMejora = cantTotalReg - cantRegConMejora;
    valorCalidadError = cantRegSinError / cantTotalReg;
    valorCalidadSospechoso = cantRegSinRegSospechosos / cantTotalReg;
    valorCalidadMejora = cantRegSinRegMejora / cantTotalReg;
    GuardarValorCalidadEnBD(probCal, cantRegSinError, cantRegSinRegSospechosos, cantRegSin-
RegMejora, cantTotalReg, valorCalidadError, valorCalidadSospechoso, valorCalidadMejora);
}
```

Cuadro 15: Cálculo de valores de calidad por problema de calidad

A continuación se detallan todos los métodos presentados en el pseudocódigo del Cuadro 15:

- *ObtenerTodosProblemasCalidad()*: obtiene todos los problemas de calidad definidos y analizados.
- *ObtenerTodasTablas(probCal)*: obtiene todas las tablas involucradas para el problema de calidad "*probCal*" pasado como parámetro.
- *ObtenerCantRegConError(probCal, tabla)*: obtiene todos los registros con error asociados a la tabla "*tabla*" pasada como parámetro y que corresponden al problema de calidad "*probCal*".
- *ObtenerCantRegSospechosos(probCal, tabla)*: obtiene todos los registros con datos sospechosos asociados a la tabla "*tabla*" pasada como parámetro y que corresponden al problema de calidad "*probCal*".
- *ObtenerCantRegConMejora(probCal, tabla)*: obtiene todos los registros con datos a mejorar asociados a la tabla "*tabla*" pasada como parámetro y que corresponden al problema de calidad "*probCal*".
- *ObtenerCantTotalReg(tabla)*: obtiene la cantidad de registros que posee la tabla "*tabla*" pasada como parámetro.
- *GuardarValorCalidadEnBD(probCal, cantRegSinError, cantRegSinRegSospechosos, cantRegSinRegMejora, cantTotalReg, valorCalidadError, valorCalidadSospechoso, valorCalidadMejora)*: agrega un registro en la tabla *cd_valor_de_calidad_por_prob_cal* con la información pasada como parámetro:
    - *probCal*: problema de calidad analizado.
    - *cantRegSinError*: cantidad de registros sin error de todas las tablas involucradas en el problema de calidad "*probCal*".
    - *cantRegSinRegSospechosos*: cantidad de registros sin datos sospechosos de todas las tablas involucradas en el problema de calidad "*probCal*".
    - *cantRegSinRegMejora*: cantidad de registros sin datos a mejorar de todas las tablas involucradas en el problema de calidad "*probCal*".
    - *cantTotalReg*: corresponde a la suma de total de registros de cada tabla involucrada en el problema de calidad "*probCal*".
    - *valorCalidadError*: valor de calidad hallado en base a los datos con error para el problema de calidad "*probCal*".
    - *valorCalidadSospechoso*: valor de calidad hallado en base a los datos sospechosos para el problema de calidad "*probCal*".
    - *valorCalidadMejora*: valor de calidad hallado en base a los datos a mejorar para el problema de calidad "*probCal*".

En el Cuadro 16, para cada clasificación de los datos se muestra los valores de calidad por problema de calidad. Se los presentan de menor a mayor valor de calidad de datos con error, dado que los registros con clasificación "Error" son más críticos en relación a los datos sospechosos o de posibles mejoras.

Problema de Calidad	Valor de Calidad de Datos con Error	Valor de Calidad de Datos Sospechosos	Valor de Calidad de Datos a Mejorar
Reglas de integridad intra-relación	0.93	0.91	1.00
Reglas de integridad referencial	0.94	1.00	1.00
Reglas de integridad de dominio	0.98	1.00	1.00
Registro Incompleto	0.99	1.00	0.99

Valor fuera de rango	1.00	0.84	1.00
Registro Inexistente	1.00	0.99	1.00
Precisión Incorrecta	1.00	1.00	0.98
Registro duplicado	1.00	1.00	1.00
Registro contradictorio	1.00	1.00	1.00

Cuadro 16: Valores de calidad por problema de calidad

A continuación se detalla los resultados obtenidos para cada problema de calidad. Durante su análisis, se descomponen las métricas involucradas y cuyo valor de calidad no es uno (a excepción del problema registro duplicado y registro contradictorio que se analizaron todas sus métricas).

6.3.1 Reglas de integridad intra-relacion

Tal como se puede apreciar en el Cuadro 16, el menor valor de calidad de datos con error corresponde al problema de calidad "Reglas de integridad intra-relación", con un valor de 0.93. Esto significa que el 7% de los registros de las tablas asociadas al mismo presentan este problema de calidad. Adicionalmente, el 9% de los demás registros son posibles candidatos a errores.

Para analizar la causa de este valor, primero se descompone el problema de calidad según sus métricas involucradas y sus valores de calidad correspondientes. En el Cuadro 17 se puede apreciar con detalle esta información, solamente para las métricas que tiene valores de calidad menores a 1.0.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Reglas de integridad intra-relación	225	La tasa de revisión de diseño estimado no puede exceder de 200 por hora LOCS	0.56	Sospechoso
	228	La tasa de revisión de código real no puede exceder de 200 por hora LOCS	0.59	Sospechoso
	227	La tasa de revisión de diseño real no puede exceder de 200 por hora LOCS	0.59	Sospechoso
	226	La tasa de revisión de código estimado no puede exceder de 200 por hora LOCS	0.60	Sospechoso
	223	No deben haber más de 5 defectos por hora en la fase de revisión de diseño	0.87	Sospechoso
	181	Dada una fase i, la fecha de inicio de dicha fase, debe ser mayor o igual a la fecha de fin de todas las fases que le anteceden en secuencia a la fase i mencionada.	0.89	Error
	224	No deben haber más de 10 defectos por hora en la fase de revisión de código	0.91	Sospechoso

	222	El tamaño total real ingresado en Planning Summary es mayor (hasta un 10%) o igual a: B - D + A & M - M + R	0.98	Error

Cuadro 17: Problema reglas de integridad intra-relación y sus métricas asociadas

Del conjunto de métricas desplegadas en el Cuadro 17, se puede apreciar claramente que las métricas 225, 226, 227 y 228, son las que afectan en mayor medida al bajo valor de calidad de datos sospechosos para el problema de calidad actual. Estas cuatro métricas fueron analizadas en la sección anterior 6.2 "Valores de calidad por métrica".

Las métricas 223 y 224 hacen referencia a la cantidad de defectos que se encuentran por hora, tanto en la fase de diseño como de codificación. En este caso, estos bajos valores de calidad se deben a que los usuarios detectan una mayor cantidad de defectos que lo sugerido para las fases mencionadas. Pero análogamente al caso de las revisiones de diseño y código, no se ve reflejado cuales son las características de los defectos que se están detectando, ni la forma en que el usuario reporta los mismos.

Las métricas 181 y la 222 son las que impactan significativamente en el valor de calidad de datos con error, esto se debe a que están clasificadas como "Error" y sus valores de calidad son 0.89 y 0.98 respectivamente. De estas dos la que tiene más peso es la 181 y por ende es la que se debe estudiar, en la sección 6.2 fue analizada y se concluye lo siguiente; al registrar una fase, ni la herramienta ni las reglas definidas sobre la base de datos de PSP validan que se hayan finalizado las fases que le preceden, además la herramienta permite registrar fases que preceden a las que ya existen.

Acciones preventivas

Sería deseable que la herramienta validara la consistencia entre las fechas de las fases de manera que no permita registrar fases con fechas de inicio y fin erróneas con respecto a las demás fases del proyecto.

6.3.2 Reglas de integridad referencial

Este problema de calidad tiene un valor de calidad de datos con error de 0.94, por lo tanto, el 6% de los registros de las tablas asociadas al mismo presentan este problema de calidad.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Reglas de integridad referencial	59	La fecha de registro PIP debe ser mayor que la fecha de inicio el proceso	0.70	Error
	180	La fecha de creación de un defecto inyectado en una fase, debe ser mayor o igual a la fecha de comienzo del proyecto.	0.89	Error
	58	Cuando se finaliza un programa es guardado como una parte. El tamaño actual de dicha parte se debe actualizar en caso de que se cambie algunos de los valores del tamaño actual del proyecto.	0.90	Error
	257	El total del tiempo que llevo remover los defectos en la fase de compilación debe ser menor al tiempo que duro dicha fase	0.91	Error
	251	La fecha en que se removió un defecto, debe estar comprendida entre la fecha de inicio y fin, de la fase en que removió dicho defecto	0.92	Error

	276	Los programas que ya cargaron los datos en la fase de PM, se consideran como programas finalizados.	0.94	Error
	258	El total del tiempo que llevo remover los defectos en la fase de test unitario debe ser menor al tiempo que duro dicha fase	0.95	Error
	72	En la sección rparts: Added la parte adicional debe tener un tipo existente (rparts Type);	0.95	Error
	22	Las fechas de comienzo de los registros de log de tiempo deben ser mayor o igual que la fecha de comienzo del programa	0.95	Error
	256	El total del tiempo que llevo remover los defectos en la fase de revisión de código debe ser menor al tiempo que duro dicha fase	0.96	Error
	57	Cuando se finaliza un programa es guardado como una parte, el valor del tamaño actual debe estar bien calculado; (B-D) + A	0.97	Error
	254	El total del tiempo que llevo remover los defectos en la fase de revisión de diseño debe ser menor al tiempo que duro dicha fase	0.99	Error

Cuadro 18: Problema reglas de integridad referencial y sus métricas asociadas

Del conjunto de métricas del problema de calidad, se puede decir que en mayor o menor medida, impactan en el valor de calidad final asociado al problema de calidad. Pero solo la métrica 59 y 180 tienen un valor inferior a 0.9. Ambas métricas fueron analizadas en la sección 6.2.

Acciones preventivas

El sistema tendría que implementar mayores controles cuándo se trata de campos de tipo fecha. El sistema tendría que avisar si la fecha de inicio de una fase es menor que la fecha de fin de la fase anterior. Y que las fechas que se manejan en todos los registros sean mayores o iguales a la fecha de comienzo del proyecto.

Para poder evitar este tipo de errores, es necesario corregir el error de los calendarios utilizados por la herramienta. Es muy importante que las fechas registradas sean exactas para una buena gestión de PSP.

6.3.3 Reglas de integridad de dominio

Este problema de calidad tiene un valor de calidad de datos con error de 0.98, lo cual significa que el 2% de los registros de las tablas asociadas al mismo presentan este problema de calidad.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Reglas de integridad de dominio	62	En la sección rparts:Base el tamaño planificado del código base para la parte debe ser mayor que 0, o igual a 0 para los casos que el tamaño actual sea mayor que 0 (no fueron	0.91	Error

		planificadas)		
	92	En la sección rparts:Reused el tamaño planificado del código reutilizado (Reused) para la parte debe ser mayor que 0 o igual a 0 para los casos que el código actual reutilizado > 0	0.96	Error
	130	En el reporte creado para el instructor, el tamaño actual de (A & M) debe ser > 0	0.96	Error
	20	La cantidad de items planificados de las partes adicionales utilizadas deben ser mayor que 0 o igual a 0 para los casos que tengan items reales (no fueron planificadas)	0.97	Error
	90	Size (Plan) debe ser > 0 o igual a 0 para los casos que Size (Actual) > 0 (no fueron planificadas)	0.98	Error
	127	En el reporte creado para el instructor, el tamaño planificado de (A & M) debe ser > 0	0.98	Error
	18	El tiempo en corregir un defecto debe ser mayor que 0	0.99	Error

Cuadro 19: Problema reglas de integridad de dominio y sus métricas asociadas

Se analiza la métrica 62 que corresponde a la de menor valor de calidad. Los errores de esta, están ligados a uno de los formularios de la herramienta, más específicamente el formulario de "PSP Size Estimating Template". La columna de nombre "Base" dentro de la sección “Plan” de "Parts:Base". El 89.5% de estos registros erróneos correspondientes a la tabla rsetbase no tienen una parte asociada (no existe relación entre rsetbase y rparts). Analizando la herramienta, esto sucede cuando se borra el nombre de la parte base. Al no contar con la funcionalidad de eliminar las partes bases, los usuarios intentan borrar manualmente generando inconsistencia en la base de datos y dejando un valor 0 para la planificación de la parte base. Para el resto de los registros, algunos errores se deben a que los usuarios se olvidaron de cargar el valor del tamaño planificado de las nuevas partes bases agregadas, y por defecto estas son 0.

Acciones preventivas

Sería conveniente que la herramienta tuviera la opción de poder eliminar las partes que se agregan ante una posible equivocación de los usuarios.

6.3.4 Registro incompleto

Este problema de calidad tiene los valores de calidad de datos con error y de datos sospechosos de 0.99% cada uno. El 1% de los registros de las tablas asociadas al mismo poseen este problema de calidad y otro 1% son candidatos a tenerlo.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Registro Incompleto	202	En la sección rparts: Added, en el sector Plan, la parte adicional debe tener un tamaño relativo (rel. Sz.) asignado	0.91	Error
	61	Completitud a nivel de registro - La propuesta debe tener la descripción del problema y de la mejora.	0.92	Mejora

	236	Los formularios PIP, deben contener una descripcion de mejora	0.93	Mejora
	89	rparts Type debe tener un valor asociado	0.97	Error
	235	Todos los defectos inyectados en las fases de Compilación o Testing, deben tener un Fix Defect asociado	0.98	Error
	73	En la sección rparts: Added la parte adicional debe tener un nombre asociado	0.98	Error

Cuadro 20: Problema de registro incompleto y sus métricas asociadas

Luego de analizar el conjunto de métricas que más afectan a este problema de calidad, se observó que para todos los casos la principal causa de que estén incompletos ciertos registros en el sistema, se debe nuevamente a la carencia de validaciones de la herramienta que no permitan que se guarden en el sistema registros que aún no están completos.

Para el caso particular de las métricas 202, 89 y 73, además de la carencia de validación de la herramienta, otra posible causa que influye en este problema, es que en el formulario de "*PSP Size Estimating Template"*, en la sección de partes agregadas, no existe la funcionalidad de borrar una parte adicional que ya fue insertada. Por lo tanto, en muchos casos, se observa que los usuarios, con la intención de eliminar dicha parte, colocan valores 0 o vacíos en todos los campos de la sección "Plan" y "Actual", de forma que no afecte las estimaciones del programa actual.

Acciones preventivas

La herramienta debería incorporar más validaciones sobre los campos obligatorios, para que no permita guardar registros incompletos. Además debería contar con la funcionalidad de eliminar partes adicionales en el formulario de "*PSP Size Estimating Template"*.

6.3.5 Valor fuera de rango

Este problema de calidad tiene un valor de calidad de datos sospechosos de 0.84, lo cual significa que el 16% de los registros de las tablas asociadas al mismo son posibles candidatos a tener este problema de calidad.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Valor fuera de rango	247	Los tiempos registrados para la fase de post mortem no superan 3 desviaciones estándar de la media	0.94	Sospechoso
	241	Los tiempos registrados para la fase de diseño no superan 3 desviaciones estándar de la media	0.96	Sospechoso
	246	Los tiempos registrados para la fase de testing unitario no superan 3 desviaciones estándar de la media	0.96	Sospechoso
	240	Los tiempos registrados durante la fase de planificación no superan 3 desviaciones estándar de la media	0.96	Sospechoso

	243	Los tiempos registrados para la fase de código no superan 3 desviaciones estándar de la media	0.97	Sospechoso
	242	Los tiempos registrados para la fase de revisión del diseño no superan 3 desviaciones estándar de la media	0.97	Sospechoso
	244	Los tiempos registrados para la fase de revisión de código no superan 3 desviaciones estándar de la media	0.98	Sospechoso
	245	Los tiempos registrados para la fase de compilación no superan 3 desviaciones estándar de la media	0.98	Sospechoso
	250	La cantidad registrada de parte reusada no supera 3 desviaciones estándar de la media	0.99	Sospechoso
	249	La cantidad registrada de parte base no supera 3 desviaciones estándar de la media	0.99	Sospechoso
	248	La cantidad registrada de nuevas partes no supera 3 desviaciones estándar de la media	0.99	Sospechoso

Cuadro 21: Problema valor fuera de rango y sus métricas asociadas

Del Cuadro 21 se desprende que todas las métricas tienen un valor de calidad entre 0.94 y 0.99, lo cual indica que siempre se detecta algún error para cada una. Para lograr entender por qué a partir de estas 11 métricas se llega a un valor total de calidad 0.84 para el problema "Valor fuera de rango", es necesario analizar en profundidad las mismas.

La métrica que tiene menor valor de calidad (0.94) para este problema es la 247, esta mide si existen registros de tiempo para la fase de "*post mortem*" que se encuentren fuera de los rangos de *[1, avg_pm+ 3*stdv_pm]* y *[1, avg_pm- 3*stdv_pm]*, donde *avg_pm* es el tiempo promedio registrado para la fase "*post mortem*" y *stdv_pm* su desviación estándar. A partir de los valores que se detectaron por esta métrica, se deriva lo siguiente:

1. El 74.7% tiene valor 0 como tiempo en la fase de "*post mortem*", al analizar y descomponer este valor se concluye:
    a. El 55 % de estos no tienen registrado tiempo para esta fase y por defecto se le asigna el valor 0.
    b. De los restantes 45%, o sea de los que sí existe registro de tiempo para esta fase, el 42% tiene el valor 0 debido a que el registro de la fecha inicio de la fase es mayor o igual que la fecha fin (la mayoría de las fechas fin tienen el registro "*1001-01-01 00:00:00*" ya explicado anteriormente). Y los otros 58%, si bien la fecha fin registrada es mayor a la de inicio, la diferencia con esta es menor a 30 segundos (< 0.5 min), por lo que el delta de tiempo que está guardado en minutos redondea y guarda 0.
2. Los restantes 25.3% se deben a que para dicha fase les llevó un tiempo que está por encima de la media al menos 3 veces su desviación estándar. Si bien para algunos usuarios este tiempo de demora es correcto, también puede suceder que en algunos casos dichas demoras sean provocadas por usuarios que tuvieron interrupciones durante la fase pero se olvidaron de registrarlas, haciendo que el tiempo no sea el adecuado.

Todas las demás métricas (241 a 246) a excepción de las 248, 249 y 250, están definidas de igual manera pero varían la fase sobre la cual están evaluando, es por eso que solo se presenta con más detalle la 247, que corresponde a la de menor valor. Para las demás se generó el Cuadro 22 a modo de resumen:

Métrica	240	241	242	243	244	245	246
% Registros con error	4	4	3	3	2	2	4
% De los registros con error que tienen un registro de tiempo 0 en la fase.	14	14.5	57.5	45.3	57.3	0	44.6
% De los registros con error que tienen un registro de tiempo que está por encima de la media al menos 3 veces la desviación estándar	86	85.5	42.5	54.7	42.7	100	55.4

Cuadro 22: Resumen análisis métricas 240 a 246

Las métricas 248, 249 y 250 no se analizan dado que su valor de calidad es de 0.99, casi óptimo.
Con la información brindada anteriormente, se deriva que para todas las fases que componen las diversas etapas de PSP, existen registros de tiempos inferiores a 1 o por encima de la media al menos tres veces su desviación estándar. Los registros de tiempos inferiores a 1, se deben a que el usuario registra una fecha de fin de fase inferior a la inicial, o que la diferencia entre estas es menor a 30 segundos. Los registros de tiempo que superan la media al menos tres veces su desviación estándar se deben a dos razones: o se olvidan de finalizar la fase cuando correspondía, o no registran las interrupciones que posiblemente tuvieron.

Como la mayoría de estas métricas involucran la misma tabla "*rpspassgtdata*" y como existen muchos errores disjuntos (no impactan la misma *tupla*), al medirlas todas juntas sobre una misma tabla su valor se incrementa hasta 0.84.

Estos errores muchas veces surgen por distracción y no por el mal uso de la herramienta. El usuario no se tendría que olvidar de finalizar una fase, ni de registrar las interrupciones que tuvieron durante el proyecto.

Acciones preventivas

No aplican acciones preventivas ya que a priori no se puede confirmar que se trate de un error.

6.3.6 Registro inexistente

Este problema de calidad tiene un valor de calidad de datos sospechosos de 0.99, lo cual significa que el 1% de los registros de las tablas asociadas al mismo posiblemente tengan este problema de calidad.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Registro Inexistente	17	Existe al menos un registro para cada una de las fases (PLAN,DLD,CODE,COMPILE,UT,PM)	0.95	Sospechoso
	232	Todos los estudiantes deben tener en sus programas, algún defecto ingresado en el log de defectos.	0.99	Sospechoso
	231	Todas las partes ingresadas por un usuario, no deben ser solamente una partes Bases o Adicionales en Size Estimating Template	0.99	Sospechoso
	229	Debe existir al menos un registro de PIP, para cada programa	0.99	Sospechoso

Cuadro 23: Problema registro inexistente y sus métricas asociadas

Para este análisis, se puede apreciar que el valor para dicho problema de calidad no alcanza una cifra tan baja en comparación con otros valores de calidad vistos anteriormente. Además, el conjunto de métricas que contribuyen a que este valor no sea el óptimo son pocas, tal como se aprecia en el Cuadro 23.

Acciones preventivas

Lo que se puede concluir nuevamente, y es un error común en las cuatro métricas, es que la herramienta carece de validaciones, permitiendo así que se inyecten errores de este estilo en el sistema. Por ejemplo, para el caso de la métrica 17, antes de que un usuario finalice un programa la herramienta debería validar si el usuario pasó por cada una de las fases que componen la etapa de PSP (PSP 0, PSP 0.1, etc.), asociada al programa actual que el usuario quiere dar por finalizado. De esta forma se evitarían este tipo de errores en la base de datos de estudio.

Lo mismo sucede con las demás métricas, la herramienta para cada programa finalizado debería verificar que haya ingresado al menos un defecto en la ventana *"Defect Log"*, y también al menos un registro correspondiente a una propuesta de mejora (PIP).

6.3.7 Precisión incorrecta

Este problema de calidad tiene un valor de calidad de datos a mejorar de 0.98, lo cual significa que el 2% de los registros de todas las tablas asociadas al tienen este problema de calidad aunque es posible corregirlos.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Precisión Incorrecta	34	La fecha que se encontró el defecto debe contemplar las horas, minutos y segundos. Nos brinda más detalle y nos ubica en el momento exacto.	0.93	Mejora

Cuadro 24: Problema precisión incorrecta y sus métricas asociadas

Se puede apreciar que para el problema de calidad Precisión Incorrecta, la única métrica que afecta al bajo valor de calidad es la que se muestra en el Cuadro 24.

Para este problema de calidad en particular, se observa que el error se debe a un error de la herramienta PSP al momento de registrar un defecto. Cuando el usuario ingresar un nuevo defecto en la pantalla "*PSP Defect Recording Log*" y que carga todos los datos requeridos (sobre todo la fecha del defecto desde el calendario), al momento de guardar los cambios la herramienta inyecta el error en el sistema. Lo que sucede es que la fecha del defecto ingresada correctamente por el usuario, se almacena en la base de datos con el formato "*dd/mm/yyyy*", perdiendo así los valores: hora, minutos y segundos, lo cual genera problemas de precisión.

La única forma que esta fecha sea almacenada correctamente con la precisión adecuada, es que el usuario luego de seleccionar desde el calendario la fecha deseada, ingrese a mano los datos referidos a la hora, minutos y segundos, lo cual resulta muy poco intuitivo para los usuarios, siendo esta la causante de que se inyecten muchos errores de precisión en el sistema.

Acciones preventivas

Como se mencionó para el problema de calidad reglas de integridad de referencial, se debe corregir el error de los calendarios utilizados por la herramienta para mejorar la precisión en las fechas.

6.3.8 Registro duplicado

Este problema de calidad tiene un valor de calidad de 1.00 para todas las clasificaciones, lo cual significa que no se encuentran errores asociados al mismo, o la cantidad de errores no es significativa.

En el Cuadro 25 se descompone el problema de calidad según las métricas que involucra, como podemos ver el valor de calidad de dichas métricas es su mayoría 1.00.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Registro duplicado	4	Las partes bases utilizadas para realizar la estimación deben ser únicas	0.97	Error
	7	Los reportes de pruebas deben ser únicos	1.00	Error
	3	Los tipos de defecto deben ser únicos	1.00	Error
	2	Los defectos deben ser únicos	1.00	Error
	1	Los registros de log de tiempo de una fase deben ser únicos	1.00	Error
	261	No se permiten estudiantes duplicados en la base de datos	1.00	Error

Cuadro 25: Problema registro duplicado y sus métricas asociadas

Es un problema de calidad del cual no se registran casi errores. Los errores encontrados son muy pocos en relación a la cantidad total de registros.

Acciones preventivas

El valor de calidad asociado a este problema de calidad es casi óptimo, a pesar de esto se detectó que la herramienta carece de validaciones de unicidad en los siguientes formularios:

- *PSP Time Recording Log*: permite duplicar el registro de fases de tiempo.
- *PSP Defect Recording Log*: permite duplicar el registro de defectos.
- *Defect Type Standard*: permite duplicar el registro de defectos.
- *PSP Size Estimating Template*: permite duplcar el ingreso de las partes utilizadas para estimar.
- *PSP Test Report*: permite duplicar el ingreso de los reportes de pruebas.
- *PSP Process Improvement Proposal*: permite duplicar el ingreso de las mejoras.

Como acción preventiva se debería agregar validaciones a cada uno de estos formularios mencionados, de manera que no permita ingresar datos duplicados.

6.3.9 Registro contradictorio

Este problema de calidad también tiene un valor de calidad de 1.00 para todas las clasificaciones, pero a diferencia del anterior está conformado por más de una métrica y hay algunos registros con error.
En el Cuadro 26 se muestran las métricas que componen el problema.

Problema de calidad	ID métrica	Definición métrica	Valor calidad métrica	Clasificación
Registro contradictorio	16	Las partes reutilizables en el proyecto deben estar sin contradicciones.	1.00	Sospechoso
	14	Las partes bases utilizadas para realizar la estimación deben estar sin contradicciones.	1.00	Sospechoso
	13	Las definiciones de tipo de defecto deben estar sin contradicciones.	1.00	Sospechoso
	12	Las definiciones de tipo de defecto deben ser consistentes	1.00	Error

Cuadro 26: Problema registro contradictorio y sus métricas asociadas

Acciones preventivas
Aplican las mismas acciones preventivas mencionadas para el problema de calidad registro duplicado.

6.4 Valores de calidad por tabla

Para obtener los valores de calidad por tabla, se calcula el porcentaje de las tuplas correctas asociadas a la tabla correspondiente. El error es a nivel de tupla, entonces para los casos de que exista más de un error sobre la misma tupla, se contabiliza como uno solo (puede ser que diferentes errores afecten diferentes campos de la misma tupla, en este caso se cuenta como uno solo). Este cálculo implementado en el script SQL se puede apreciar mejor en el pseudocódigo que se presenta en el Cuadro 27.

```
listadoTablas = ObtenerTodasTablasEnMetricas();
Para cada tabla en listadoTablas
{
    cantRegConError = ObtenerCantRegConError(tabla);
    cantRegSospechosos = ObtenerCantRegSospechosos(tabla);
    cantRegConMejora = ObtenerCantRegConMejora(tabla);
    cantTotalReg = ObtenerCantTotalReg(tabla);
    cantRegSinError = cantTotalReg - cantRegConError;
    cantRegSinRegSospechosos = cantTotalReg - cantRegSospechosos;
    cantRegSinRegMejora = cantTotalReg - cantRegConMejora;
    valorCalidadError = cantRegSinError / cantTotalReg;
    valorCalidadSospechoso = cantRegSinRegSospechosos / cantTotalReg;
    valorCalidadMejora = cantRegSinRegMejora / cantTotalReg
    GuardarValorCalidadEnBD(tabla, cantRegSinError, cantRegSinRegSospechosos, cantRegSinReg-
Mejora, cantTotalReg, valorCalidadError, valorCalidadSospechoso, valorCalidadMejora);
}
```

Cuadro 27: Cálculo de valores de calidad por tabla

A continuación se detallan todos los métodos presentados en el pseudocódigo del Cuadro 27:

- *ObtenerTodasTablasEnMetricas()*: obtiene todas las tablas que son analizadas por las métricas.
- *ObtenerCantRegConError(tabla):* obtiene la cantidad de registros con error de la tabla pasada como parámetro.
- *ObtenerCantRegSospechosos(tabla):* obtiene la cantidad de registros con datos sospechosos de la tabla pasada como parámetro.
- *ObtenerCantRegConMejora(tabla):* obtiene la cantidad de registros con datos a mejorar de la tabla pasada como parámetro.
- *ObtenerCantTotalReg(tabla):*obtiene la cantidad de registros que contiene la tabla pasada por parámetro.
- *GuardarValorCalidadEnBD(tabla, cantRegSinError, cantRegSinRegSospechosos, cantRegSinRegMejora, cantTotalReg, valorCalidadError, valorCalidadSospechoso, valorCalidadMejora):* agrega un registro en la tabla *cd_valor_de_calidad_por_tabla* con la información pasada como parámetro:
    - *tabla*: tabla analizada.
    - *cantRegSinError*: cantidad de registros sin error de todas las tablas involucradas en el problema de calidad "*probCal*".

- *cantRegSinRegSospechosos*: cantidad de registros sin datos sospechosos de todas las tablas involucradas en el problema de calidad "*probCal*".
- *cantRegSinRegMejora*: cantidad de registros sin datos a mejorar de todas las tablas involucradas en el problema de calidad "*probCal*".
- *cantTotalReg*: corresponde a la suma de total de registros de cada tabla involucrada en el problema de calidad "*probCal*".
- *valorCalidadError*: valor de calidad de errores hallado para la tabla "*tabla*".
- *valorCalidadSospechoso*: valor de calidad de datos sospechosos hallado para la tabla "*tabla*".
- *valorCalidadMejora*: valor de calidad de datos con mejora hallado para la tabla "*tabla*".

El Cuadro 28 muestra los valores de calidad por tabla.

Tabla	Valor de Calidad de Datos con Error	Valor de Calidad de Datos Sospechosos	Valor de Calidad de Datos a Mejorar
rpips	0.70	0.99	0.92
rlogtdetail	0.78	1.00	1.00
rpspassgtdata	0.80	0.38	1.00
rprojects	0.83	0.92	1.00
rsetadd	0.89	0.99	1.00
rparts	0.90	1.00	1.00
rsetbase	0.91	1.00	1.00
rsetreuse	0.96	1.00	1.00
rlogddetail	0.97	0.99	0.92
rprogramsize	0.98	1.00	1.00
rtasks	1.00	1.00	1.00
rtaskscheduleplans	1.00	1.00	1.00
rtestreports	1.00	1.00	1.00
ruserprofiles	1.00	1.00	1.00
rscheduleweeks	1.00	1.00	1.00
rdefecttype	1.00	1.00	1.00
rprocessphase	1.00	1.00	1.00
rprocesses	1.00	1.00	1.00
rphases	1.00	1.00	1.00
rphasedata	1.00	1.00	1.00
rparttypestandard	1.00	1.00	1.00
rusers	1.00	1.00	1.00

Cuadro 28: Valores de calidad por tabla

Se realiza un análisis de los cuatro valores de calidad de error más bajos, identificando cuáles son las métricas que influyen en cada caso. El análisis de las métricas con menor valor de calidad fue realizado en la sección 6.2. De esta manera podemos ver cómo las métricas con menores valores de calidad afectan los valores de calidad por tabla.

Las métricas involucradas en cada una de las cuatro tablas que se presentan a continuación, se encuentran detalladas en el Cuadro 28, ver ANEXO E "*VALORES DE CALIDAD POR TABLA"*.

- **Tabla**: rpips
  Descripción de la tabla: Se registran todas las propuestas de mejoras del proceso desde el template "*Process Improvement Proposal*".

Cantidad de métricas: 5
Valor de calidad:

Valor de Calidad de Datos con Error	Valor de Calidad de Datos Sospechosos	Valor de Calidad de Datos a Mejorar
0.70	0.99	0.92

Para el caso de la tabla "rpips", tal como se puede apreciar en el *ANEXO E*, el número de métricas involucradas con dicha tabla son solamente 5.
El valor de calidad de datos con error para la tabla "rpips" es de 0.70, este valor surge directamente de la métrica 59 ("La fecha de registro PIP debe ser mayor que la fecha de inicio del programa") dado que es la única métrica que clasifica error en esta tabla. El análisis de la métrica se puede apreciar en la sección 6.3.1.

- **Tabla**: rlogtdetail
  Descripción de la tabla: Se registran todos los tiempos de inicio y fin de cada fase, desde el template "*PSP Time Recording Log*" de la herramienta.
  Cantidad de métricas: 9
  Valor de calidad:

Valor de Calidad de Datos con Error	Valor de Calidad de Datos Sospechosos	Valor de Calidad de Datos a Mejorar
0.78	1.00	1.00

En este caso observamos que la métrica 181 ("Dada una fase *i*, la fecha de inicio de dicha fase debe ser mayor o igual a la fecha de fin de todas las fases que le anteceden en secuencia a la fase *i*") es la que tiene mayor influencia en el valor de calidad de datos con error de la tabla, ya que las demás tienen valores por encima de 0.90. El análisis de la métrica se puede apreciar en el punto 6.1.

- **Tabla**: rpspassgtdata
  Descripción de la tabla: Se registra un resumen con los principales valores de todas las tablas, asociadas a los proyectos de los usuario, luego que estos acceden a la sección "*import/export*" y almacenan los datos y estructuras desde la herramienta.
  Cantidad de métricas: 25
  Valor de calidad:

Valor de Calidad de Datos con Error	Valor de Calidad de Datos Sospechosos	Valor de Calidad de Datos a Mejorar
0.80	0.38	1.00

Para el caso de la tabla "rpspassgtdata", se puede observar en el ANEXO E que de las 25 métricas involucradas a la misma, 11 corresponden a métricas que clasifican errores y las restantes clasifican datos sospechosos. Esto quiere decir que esas 11 métricas determinan que el valor de calidad de datos con error sea 0.80 y las restantes 14 el valor de calidad de calidad de datos sospechosos de 0.38.
Para el caso de las métricas que clasifican errores, 5 poseen el valor de calidad 1 y las restantes entre 0.91 y 0.99. Y para las métricas que clasifican datos sospechosos se puede observar que las métricas 225, 228, 227 y 226 (en ese orden), son las grandes responsables del bajo valor de calidad, se destacan del resto debido a que tiene valores de calidad inferiores a 0.60. Las mismas ya fueron analizadas en la sección 6.1.

- **Tabla**: rprojects
  Descripción de la tabla: Se registran los datos generales e indicadores actuales asociados a cada uno de los proyectos.
  Cantidad de métricas: 8
  Valor de calidad:

Valor de Calidad de Datos con Error	Valor de Calidad de Datos Sospechosos	Valor de Calidad de Datos a Mejorar
0.83	0.92	1.00

En este caso también hay un conjunto de métricas responsables del valor de calidad de la tabla, pero la métrica 180 ("la fecha de creación de un defecto inyectado en una fase, debe ser mayor o igual a la fecha de comienzo del proyecto"), es la que más se diferencia de las demás, la cual ya fue analizada en el punto 6.1.

Como se puede ver en los cuadros anteriores, el valor de calidad por métrica influye directamente en los valores de calidad por tabla, ya que cada métrica tiene lo que se denomina una tabla "activa", que es la tabla principal medida por la métrica.

Los valores de calidad por tabla pueden ser más bajos que el menor valor de calidad de las métricas que afectan a la tabla, ya que los errores detectados por las distintas métricas pueden ser encontrados en tuplas diferentes. Esto se aplica a todas las tablas del sistema.

6.5 Análisis transversal

En las secciones anteriores se analizaron los valores de calidad asociados a las métricas, tablas y problemas calidad. Estos enfoques son muy importantes y se complementan entre sí, pero se pueden obtener otros valores significativos que se desprenden de analizar la *metadata* generada y no directamente de los valores de calidad encontrados. Por ejemplo, resulta interesante conocer los usuarios que cometen mayor cantidad de errores al usar la herramienta, y donde los cometen (tablas, formularios, métricas, entre otras). A través de una consulta *SQL* se pueden obtener estos resultados, el Cuadro 29 muestra los usuarios que superan los 1000 errores.

Identificador Usuario	Cantidad de datos con error	Cantidad de datos sospechoso	Cantidad de datos a mejorar
12	2455	32	0
171	2247	32	0
118	1732	18	9
229	1417	16	8
238	1254	23	2
197	1103	19	4

Cuadro 29: Usuarios que superan los 1000 errores

En el Cuadro 29 se aprecia que existen seis usuarios que superan los 1000 errores. O sea, que al desarrollar los programas utilizando la herramienta de PSP, estos usuarios generan un total de 10208 errores, 140 datos sospechosos y 23 datos a mejorar.

El cuadro completo se encuentra en el ANEXO G *"Errores por usuario"*, del mismo se desprenden otros datos relevantes:

- De los 456 usuarios registrados en el sistema que implementaron siguiendo el proceso PSP, cada uno genera al menos 4 registros con error.
- En promedio cada usuario genero 131 registros con error, 9 datos sospechosos y 19 datos a mejorar.

Estas cantidades mencionadas se pueden conocer en detalle con el desglose de las métricas involucradas en cada error. Recordar que se considera un error a un registro que contiene un problema de calidad, ya sea por la existencia de un error en los datos, o por la identificación de una oportunidad de mejora, o de un valor sospechoso para alguna de las tablas involucradas en este estudio de calidad.

En el ANEXO G se presenta la información de los tres usuarios más problemáticos. Del análisis de sus valores se detecta que el 94.8% de los errores se desprenden de la métrica 181 (“Encontrar un programa para el cual exista una fase i y una fase j, talque la fecha de inicio de la fase i sea mayor a la fecha fin de la fase j, con fase j posterior en secuencia a la fase i”). En la sección 6.2 se presentaron los motivos por los cuales se generaron estos errores (tanto por la falta de controles por parte de la herramienta como por la distracción de los usuarios al momento de registrar sus fases).

7 Conclusiones y trabajos a futuro

Durante este proyecto se utilizó un enfoque sistemático, disciplinado y estructurado, aplicando conceptos y técnicas de la disciplina Calidad de Datos para identificar y medir los problemas de calidad en los datos recolectados durante el uso del PSP.

La detección de errores en los datos es importante en el contexto de aplicar procesos de desarrollo de software. Los datos que se generan durante el uso de un proceso son utilizados luego para predicciones y cálculos de avances de proyecto, para generar información histórica, entre otros. Si los datos que se utilizan son de mala calidad, puede suceder que las decisiones que se tomen durante el proyecto, o durante uno nuevo que utilice dicha información, sean equivocadas.

Los pasos que se siguieron en el transcurso de este proyecto, hasta obtener el modelo inicial de calidad de datos y los valores de calidad consistieron en: identificar los posibles errores que pueden suceder sobre los datos, a partir del conocimiento de las herramientas utilizadas, de las bases de datos generadas y de la temática bajo estudio. Los errores se categorizan según dimensión, factor, y problema de calidad, y son estos conceptos los que empiezan a definir como se formará el modelo de calidad. A partir de estos posibles errores se definen las métricas que se utilizarán para medir la calidad, y conformar el modelo inicial junto con las dimensiones, factores y problemas de calidad. Luego se procede a implementar los métodos de medición correspondientes. Finalmente se presentan los valores de calidad, previamente se genera la *metadata* que contiene la información necesaria (ubicación y descripción) sobre los errores encontrados, y es en base a esta que se obtienen dichos valores.

7.1 Conclusiones

El proceso de desarrollo de software definido por PSP permite mejorar la productividad de las personas y perfeccionar los hábitos de programación. Se puede lograr una detección temprana de defectos, lo que permite mitigar riesgos y ayuda a cumplir con los objetivos de los proyectos.

PSP puede ser muy efectivo si se realiza un registro detallado de la información generada en cada proyecto y con apoyo de un plan con base a estimaciones certeras. Todo esto es posible si los datos que se van generando son correctamente almacenados. Es aquí donde la herramienta que fue analizada en este proyecto tiene gran importancia, dado que permite generar la información necesaria y obtener retroalimentación de manera transparente y sencilla. La robustez y la correctictud de la herramienta es un punto crítico para que PSP cumpla sus objetivos.

Desde la perspectiva de la calidad de datos, este estudio muestra la aplicación de técnicas de medición de calidad de datos a un dominio particular (PSP). Luego de analizar un conjunto de dimensiones y factores de calidad de datos, se seleccionaron aquellos que aplicaban a este dominio de estudio.

En la siguiente gráfica, se muestran las dimensiones y factores estudiados y cuál fue la aplicación de cada uno (cantidad de métricas definidas):

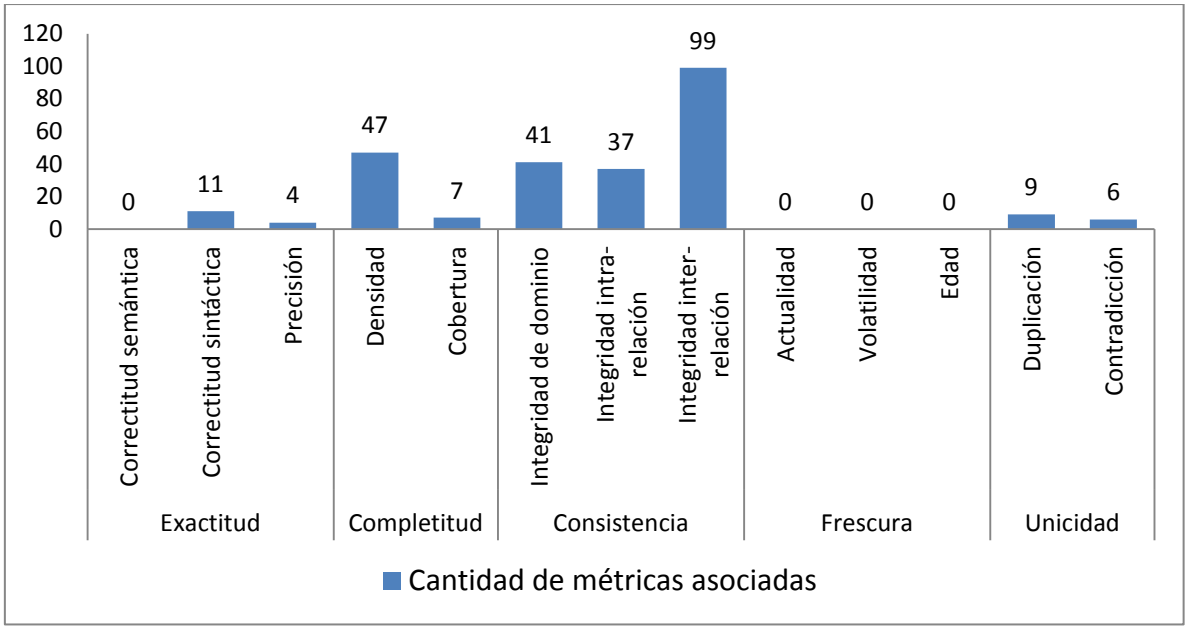

Figura 10 - Cantidad de métricas asociadas a cada factor de calidad analizado

Por lo que se puede apreciar en la gráfica de la Figura 10 el modelo inicial de calidad está definido por las dimensiones: exactitud, completitud, consistencia y unicidad. Los factores de calidad que conforman el modelo son: correctitud sintáctica, precisión, densidad, cobertura, integridad de dominio, integridad intra-relación, integridad inter-relación, duplicación y contradicción.

Al tratarse de una base de datos generada exclusivamente para este tipo de estudio, estamos hablando de una base de datos estática, que no tiene actualización de información en tiempo real, y que los datos fueron cargados una única vez, por lo tanto no se encontraron métricas para ninguno de los factores de la dimensión frescura. Al contrario la dimensión consistencia fue la que más métricas registró, ya que incluye control de dominio sobre cada atributo y control de integridad inter-relación (que incluye a los controles de integridad referencial).

Desde la perspectiva del PSP, este proyecto ayuda a comprender dónde se encuentran las mayores dificultades a la hora de uso de la herramienta y comprensión del proceso. A partir de los errores encontrados y almacenados en la *metadata*, se puede obtener información relevante como por ejemplo; pantallas en las cuales se introducen errores, usuarios que registran mayor cantidad de problemas, tablas en las cuales se encuentran la mayor cantidad de registros con error. La *metadata* generada es muy valiosa ya que permite obtener información variada como ubicación y descripción de los errores, valores de calidad, entre otros.

A partir de la *metadata*, los errores pueden ser completamente identificados (por tabla, fila, atributo(s) y error). Al estar identificados de esta forma, resulta de gran ayuda a la hora de realizar una limpieza y/o migración de datos de la base.

Como resultado del trabajo de calidad en la base de datos de PSP se obtienen valores de calidad, agrupados de acuerdo a tres grupos de análisis: por problema de calidad, por tabla y por métrica. A su vez para los valores de calidad por problema de calidad y por tabla, se separan los valores en los que son errores (registros donde existe la certeza que contienen error), sospechosos (registros que son candidatos a ser errores pero que no existe la seguridad para poder afirmarlo) y mejoras (son registros en los cuales surgen oportunidades de mejora). Esta clasificación se realiza ya que el valor de cada problema y cada tabla lo aportan varias métricas, y no todos los errores detectados por cada métrica tienen el mismo impacto en la calidad. Para los valores de calidad por métrica no es necesario hacer esta distinción, ya que la granularidad es la propia métrica.

La siguiente gráfica muestra la cantidad de métricas por rango de valores de calidad:

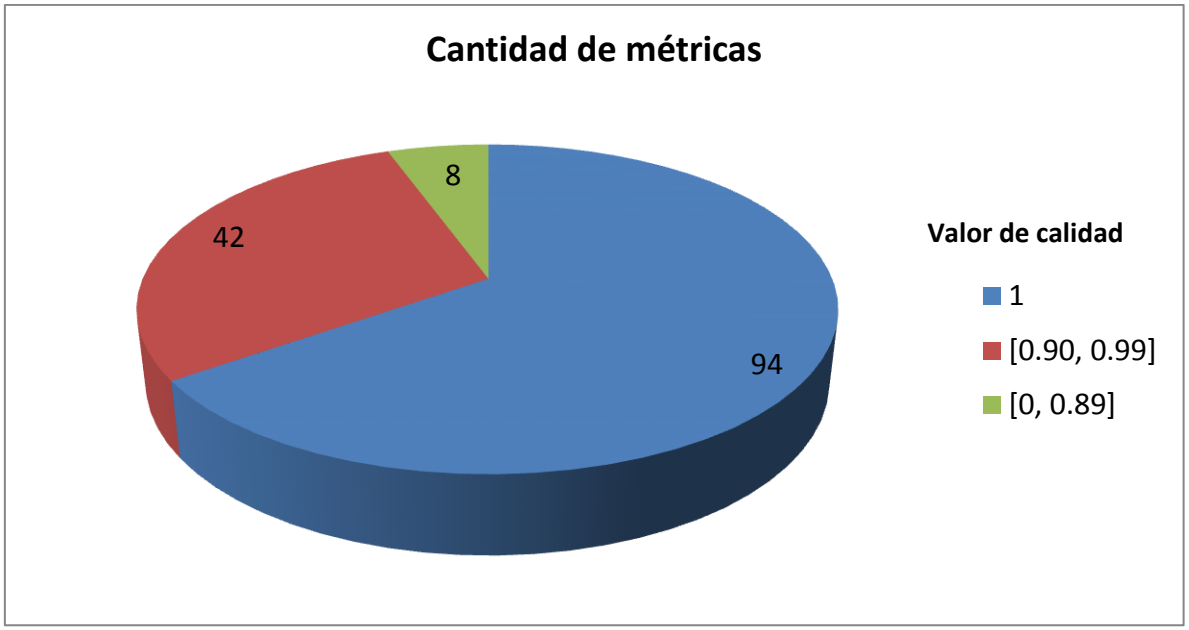

Figura 11 - Cantidad de métricas por rango de valores de calidad

La Figura 11 indica que el 65.3% de las métricas no detectan errores. El 29.2% poseen un valor de calidad entre 0.90 y 0.99, lo que implica que no encuentran muchos errores. El restante 5.5% tienen un valor de calidad inferior a 0.89. Por lo que a nivel de métrica se puede afirmar que el valor de calidad es bueno.

A nivel de tabla se tienen los siguientes valores de calidad:

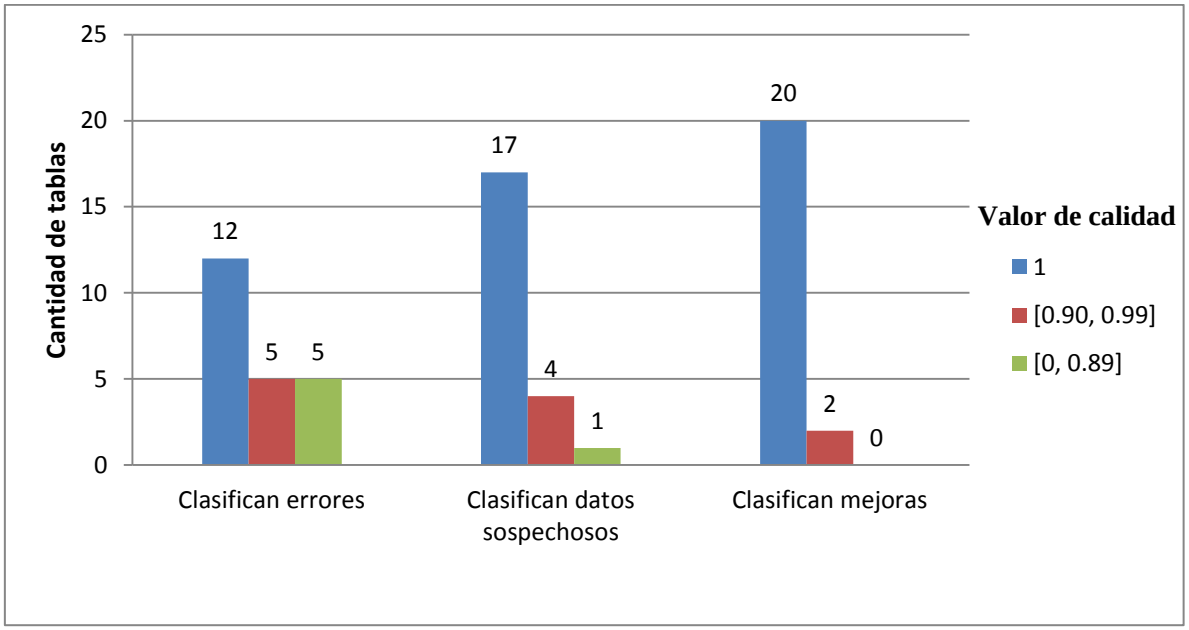

Figura 12 - Cantidad de tablas por rango de valores de calidad

Como se aclaró anteriormente los valores de calidad por tabla se separan en: errores, sospechosos y mejoras. Si bien más del 50% de los valores de calidad tienen valor igual a uno, la cantidad de errores ya no es tan despreciable como en los valores de calidad por métrica. Esto es porque varias métricas inciden en el valor de calidad de cada tabla, entonces si bien el valor de calidad por métrica puede ser alto, al sumarse varias métricas que afectan una tabla, la calidad de la misma puede no ser tan buena.

A nivel de problema de calidad se tienen los siguientes valores de calidad:

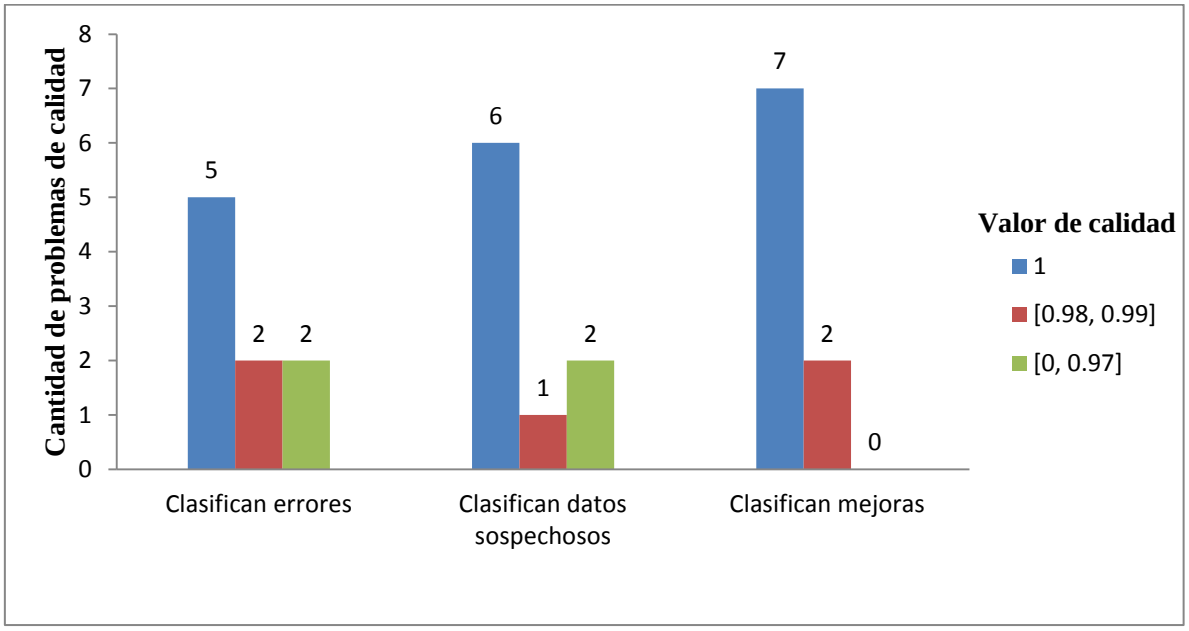

Figura 13 - Cantidad de problemas de calidad por rango de valores de calidad

Los valores de calidad por problema de calidad se clasifican de la misma manera que los valores de calidad por tabla: en errores, sospechosos y mejoras. En este caso ocurre algo similar a los valores de calidad por tabla. Varias métricas aportan el valor de calidad para un problema de calidad, entonces si bien los valores de calidad por métrica son buenos, al sumarse todas las métricas para cada problema de calidad su valor puede no ser tan bueno.

Del análisis de los valores de calidad, se detectaron algunas mejoras y acciones preventivas que se pueden tomar con respecto a la herramienta "*PSP Student WorkBook*". Si bien todos los formularios de la herramienta son importantes, uno de los más decisivos para generar la estimación es *PSP Size Estimating Template*. En el análisis de los valores de calidad por métrica (ver sección 6.2 "*Valores de calidad por métricas"*) se sugieren algunas mejoras deseables para este formulario.

Una de las mayores dificultades durante la realización de este proyecto se encontró a la hora de hacer la unión de las métricas encontradas en este proyecto, con las métricas del estudio [14]. La búsqueda de métricas repetidas insumió mucho esfuerzo, debido a la diferencia de criterios de clasificación de métricas, semántica y hasta en el lenguaje natural utilizado para describirlas. El total de métricas encontradas (factibles y no factibles) en este proyecto fueron 158, en el estudio [14] se encontraron 107. Se encontró que había 104 métricas en común.

A nivel personal, este estudio nos brindó la posibilidad de profundizar nuestro conocimiento en dos grandes áreas, como son Calidad de Datos y Procesos de Desarrollo de *Software*. En la primera aprendiendo a identificar y aplicar los aspectos fundamentales en un estudio de Calidad de Datos y desarrollar una forma sistemática y estructurada a seguir para implementar estudios de este tipo. En el área de procesos, aprendiendo una metodología eficiente para la mejora personal en los procesos de desarrollo de software de un ingeniero.

7.2 Trabajos a futuro

Una continuación natural de este trabajo sería la limpieza de datos, y la migración a una nueva base de datos (si corresponde). La *metadata* generada, producto de crear la estructura para almacenar los resultados de las métricas de calidad, fue pensada de tal manera que sean fácilmente identificables las *tuplas* con

error, teniendo información sobre la tabla, la clave, tipo de error y categorización del mismo. Esto hace posible que a la hora de la limpieza se tenga bien identificado el registro, y que se tenga claro que acción se debe tomar con el mismo. Por lo que la *metadata* generada por este estudio de calidad sería de gran ayuda para una limpieza de datos.

En el proceso de limpieza de datos se puede detectar que la estructura de la base que los contiene no es la adecuada, por lo que se puede proceder a la reestructura de la misma, de manera que se eliminen inconsistencias detectadas en el diseño o se agreguen mejoras para una mejor calidad de datos.

Una vez llevada a cabo la limpieza y posible reestructura de la base, sería interesante un trabajo en el cual se comparen los resultados de la ejecución de las métricas de medición de la calidad sobre la base de datos original y sobre la nueva base de datos. Esta comparación daría una idea del impacto de la limpieza y la influencia de la mala calidad de los datos sobre el valor de calidad de la base.

También a partir de todos los errores encontrados y posibilidades de mejora, se puede mejorar la herramienta PSP Student Workbook. Ya que con los datos que se encuentran en la metadata se puede obtener las pantallas en las cuales se tienen mayores dificultades, o donde los usuarios cometen más errores.

Debido a que en este trabajo se construye un modelo *inicial* de calidad de datos, un trabajo a futuro que resulta fundamental es lograr generalizar la propuesta de forma que el modelo de calidad y la metodología de trabajo puedan ser aplicadas sobre otras bases de datos de PSP.

8 Glosario

- Métrica: es un instrumento que define la forma de medir un factor de calidad.
- Problema de calidad: decimos que existe un problema de calidad sobre los datos de PSP cuando no se cumple con alguna de las métricas de calidad definidas, las cuales están basadas en los conceptos de la teoría de calidad de datos.
- Búsqueda: de problemas de calidad en los datos. Búsqueda de lugares que pueden ser causa de ingreso de información de mala calidad en el sistema.
- Estudio externo: Un estudio de la calidad de los datos recolectados durante el uso del *Personal Software Process* (Carolina Valverde, Fernanda Grazioli, Diego Vallespir). Facultad de Ingeniería – Universidad de la República – Año 2012. Referencia [14].
- Valor de calidad: el valor de calidad es un número que indica para una tabla el porcentaje de registros sin error que posee. El mismo se calcula como: (cantidad de tuplas sin error) / (cantidad de tuplas totales)
- Error: es la instanciación de un problema de calidad sobre un atributo, tabla o un conjunto de estos. Se refiere a un error en la calidad de los datos.
- MySQL: es un sistema de gestión de base de datos relacional, multihilo y multiusuario.

9 Referencias

1. C. Batini and M. Scannapieco, Data quality: concepts, methodologies and techniques. Springer, 2006.
2. A. Marotta, "Calidad de Datos." Instituto de Computación, Facultad de Ingeniería de la UdelaR, Montevideo, 2010.
3. M. Sánchez, J. Mora and J. Alonso, "Tutorial: Introducción a la Herramienta PSP Student WorkBook", Software Engineering Lab (SEL), Universidad Carlos III de Madrid.
4. The Standish Group, "The CHAOS Report", 2001.
5. Software Engineering Institute (SEI), Carnegie Mellon University, [Online], Available: HYPERLINK "http://www.sei.cmu.edu" http://www.sei.cmu.edu
6. M. Paulk, C. Curtis, M. Chrissis and C. Weber, *Capability Maturity Model (CMM)*, Febrero 1993, [Online], Available: HYPERLINK "http://www.sei.cmu.edu/reports/93tr024.pdf" http://www.sei.cmu.edu/reports/93tr024.pdf
7. S. Bayona, J. Calvo, G. Cuevas and T. San Feliu, "Team Software Process (TSP): Mejoras en la estimación, calidad y productividad de los equipos en la gestión de software", Facultad de Informática Universidad Politécnica de Madrid Campus de Montegancedo, Enero 2007.
8. W. Humphrey and T. Chick, "Team Software Process (TSP) Body ofKnowledge (BOK)", Software Engineering Institute (SEI), Carnegie Mellon University.
9. E. Larco, "Uso del PSP (Personal Software Process) en el desarrollo de software, Quito, Marzo 2007, [Online], Available: HYPERLINK "http://bibdigital.epn.edu.ec/bitstream/15000/345/1/CD-0760.pdf" http://bibdigital.epn.edu.ec/bitstream/15000/345/1/CD-0760.pdf
10. Introducción al PSP (Personal Software Process), 2010, [Online], Available: HYPERLINK "http://www.uv.mx/personal/asumano/files/2010/07/PSP.pdf" http://www.uv.mx/personal/asumano/files/2010/07/PSP.pdf
11. S. Moreno, L. Perez, F. Grazioli and D. Vallespir, "Principios y fundamentos del Proceso Personal de Software" Instituto de Computación, Facultad de Ingeniería de la UdelaR, Montevideo 2008.
12. W. Humphrey, *PSP: A Self-improvement Process for Software Engineers*, First. Addison-Wesley Professional, 2005
13. E. Rahm and H. Do, "Data cleaning: Problems and current approaches," IEEE Data Eng. Bull., 2000.
14. C. Valverde, F. Grazioli, and D. Vallespir, "Un Estudio de la Calidad de los Datos Recolectados durante el Uso del Personal Software Process," in JIISIC 2012, 2012
15. C. Valverde and B. Bianchi, "Un caso de estudio en Calidad de Datos para Ingeniería de Software Empírica (Tesis de Grado)," Universidad de la República, Uruguay, 2009.
16. D. Strong, Y. Lee, and R. Wang, "Data Quality in Context," *Commun. ACM*, vol. 40, no. 5, pp. 103–110, 1997.
17. T.C. Redman, *Data Quality for the Information*, Age 1st ed. Norwood, MA, USA: Artech House, Inc., 1997.
18. M. Scannapieco and T. Catarci, "Data quality under a computer science perspective," in *Archivi & Computer*, 2002, pp. 1-12.
19. C. Valverde, "Calidad de Datos en Experimentos de Ingeniería de Software," Universidad de la República, Uruguay, diciembre 2014.
20. P. Johnson and A. Disney, "A critical analysis of PSP data quality: Results from a case study," in Empirical Software Engineering, 1999, vol. 349, pp. 317–349.