FastWorks FPA: Una herramienta para automatizar la medición del tamaño funcional

Christian Quesada-López, Marcelo Jenkins
CITIC, Universidad de Costa Rica
San Pedro, Costa Rica
{cristian.quesadalopez, marcelo.jenkins}@ucr.ac.cr

Luis Carlos Salas, Juan Carlos Gómez
Grupo Asesor en Informática
San José, Costa Rica
{lsalas, jgomez}@grupoasesor.net

Resumen—La automatización de la medición del tamaño funcional es un reto para la industria del *software*. Una solución automatizada de conteo no es trivial, debido a que debe proporcionar un nivel aceptable en la exactitud de las mediciones de acuerdo a los estándares de la industria. En este artículo se presenta una herramienta prototipo para la medición del tamaño funcional para aplicaciones de *software* modeladas en un marco de trabajo para desarrollo llamado FastWorks. El estudio detalla la arquitectura del maco de trabajo y del componente de medición, el diseño de las reglas de mapeo y conteo del procedimiento de medición basado en el método IFPUG FPA y la validación de los resultados del conteo de puntos de función obtenidos por la herramienta.

Palabras clave—medición del tamaño funcional; puntos de función IFPUG; procedimiento de medición; estimación del tamaño funcional

I. INTRODUCCIÓN

En la planificación de proyectos, la medición del tamaño funcional (FSM) del *software* es fundamental [1]. La FSM puede ser utilizada para la estimación del esfuerzo y el costo de los proyectos, para la evaluación de la calidad, para el monitoreo y control de los avances, para negociaciones de cambios y para la medición de la productividad y la densidad de defectos. Además, el tamaño funcional se considera una métrica base para los programas de medición en las organizaciones de desarrollo de *software* ya que permite generar una variedad de indicadores de productividad, calidad y finanzas en distintas fases del proceso de desarrollo [2]. En la actualidad existen cinco métodos estandarizados: COSMIC FFP (ISO/IEC 19761), IFPUG FPA (ISO/IEC 20926), MkII (ISO/IEC 20968), NESMA (ISO/IEC 24570) y FiSMA (ISO/IEC 29881), donde el método IFPUG FPA y COSMIC FFP son los más utilizados en la industria [1, 3].

Las principales limitantes en la adopción de los métodos FSM son el tiempo y el costo del proceso de conteo [4]. El cálculo de métricas funcionales generadas a partir de marcos específicos de trabajo para el desarrollo del *software* permite obtener beneficios tales como la reducción del tiempo y costo del proceso de conteo y la mejora de la consistencia de los resultados de medición. El reto en la automatización del proceso de conteo es proporcionar mediciones confiables y reducir la variabilidad de los resultados para que sean aceptables en la industria [5]. Es necesaria evidencia empírica sobre la automatización de la medición para incrementar su aplicación en la industria [5, 6] y es requerida una validación rigorosa y auditable de los procedimientos que garantice la calidad de los resultados del proceso de medición [7]. Las herramientas que han sido desarrolladas para realizar el conteo automatizado de puntos de función aun no proveen los resultados requeridos por la industria [8].

En este estudio se presenta una herramienta para la medición del tamaño funcional del *software* basada en el método IFPUG FPA. La herramienta permite medir automáticamente aplicaciones modeladas en un marco de trabajo para el desarrollo de *software* llamado FastWorks (FW), el cual ha sido implementado internamente por una organización de desarrollo de *software* costarricense durante más de 15 años para generar aplicaciones transaccionales. El artículo detalla la arquitectura del maco de trabajo y del componente de medición, y presenta el procedimiento de medición y la validación de los resultados del conteo de puntos de función. El objetivo es proveer la posibilidad de medir automáticamente el tamaño funcional de las aplicaciones modeladas y desarrolladas por la organización en su marco de trabajo.

Las secciones restantes del artículo son las siguientes: la Sección II describe los conceptos básicos sobre la medición del tamaño funcional, la Sección III presenta los trabajos relacionados. La Sección IV describe la herramienta y procedimiento de medición, la Sección V presenta los resultados preliminares de la validación y finalmente, la Sección VI presenta las conclusiones y el trabajo futuro.

II. MEDICIÓN DEL TAMAÑO FUNCIONAL

A. *Medición de puntos de función*

El análisis de puntos de función (IFPUG FPA) [9] es una metodología utilizada para medir el tamaño funcional de una aplicación de *software* que se basa en la propuesta creada originalmente por Allan Albrecht en 1979 [10]. Esta mide la funcionalidad entregada al usuario independientemente de su implementación. IFPUG FPA es una de las principales métricas de tamaño funcional en la industria donde gobiernos como Brasil exigen actualmente en sus contratos esta métrica y otros como Corea del Sur e Italia podrían seguir esta tendencia [11].

Este método analiza los requerimientos, transacciones y datos de una aplicación que sean significativos para el usuario final, clasifica cada una de las funciones en cinco componentes básicos (BFC) y los cuenta de acuerdo a criterios de complejidad predefinidos. Los BFC son mapeados a valores numéricos y la suma de ellos constituye el conteo de tamaño funcional. Los BFC pueden ser clasificados en archivos lógicos internos (ILF) e interfaces externas (EIF) llamados funciones de datos (DF), y entradas (EI), salidas (EO) y consultas (EQ) llamados funciones transaccionales (TF). El tamaño funcional se determina calculando la complejidad de cada uno de los componentes funcionales básicos (BFC). Para

las DF, la complejidad se calcula a partir de la cantidad de elementos de datos (DET) y elementos de registro (RET), para esto utiliza una matriz de complejidad (RET/DET). Del mismo modo, para las TF, la complejidad se calcula a partir de la cantidad de DET y archivos referenciados (FTR), para esto utiliza una matriz de complejidad (FTR/DET). Un DET es un campo de datos no recursivo, único y reconocible por el usuario, un RET es un subgrupo de elementos de datos reconocible por el usuario y un FTR es un archivo lógico interno leído o mantenido, o externo leído, por una función. La suma de todos los valores corresponde al tamaño funcional sin ajustar (UFP).

Los pasos para realizar el conteo de una aplicación con el método IFPUG FPA son los siguientes: (1) determinar el tipo de conteo y reunir la documentación y artefactos disponibles, (2) identificar el alcance del conteo, los límites de la aplicación y los requerimientos funcionales de los usuarios, (3) identificar y contar las funciones de datos, (4) identificar y contar las funciones transaccionales, (5) realizar el conteo de puntos función sin ajustar, (6) determinar el factor de ajuste de acuerdo a las características generales de la aplicación y (7) calcular el conteo del tamaño funcional ajustado. Los detalles para la aplicación del método pueden encontrarse en [9].

B. Procedimientos de medición del tamaño funcional

Un método de medición, tal como IFPUG FPA, es una secuencia lógica de operaciones descritas de manera general. Un procedimiento de medición presenta una descripción detallada y un conjunto de operaciones, descritas específicamente, para obtener una medición conforme a uno o más principios de medida y a un método de medición de tamaño funcional específico. Un procedimiento de medición de tamaño funcional puede seguir los pasos del modelo de procesos para métodos de medición del tamaño funcional propuesto por Abran [12]. Este requiere la identificación de un número de elementos y la aplicación de reglas específicas para asegurar que el resultado de una medición cumple con un conjunto de criterios de calidad y que se realiza una correcta aplicación de las reglas de mapeo y conteo del procedimiento.

El modelo de procesos [12] se divide en tres etapas bien definidas: (1) el diseño de la medición (el principio de medición, el método de medición y el procedimiento de medición), (2) la aplicación del método de medición en un contexto determinado mediante el procedimiento de medición para producir un resultado del tamaño funcional, y (3) la explotación de los resultados de medición utilizando modelos de predicción en un contexto determinado. En la práctica, las actividades de verificación se realizan en cada una de las tres etapas para garantizar el suficiente grado de confiabilidad. La herramienta propuesta en este estudio implementa un procedimiento de medición que fue diseñado siguiendo las etapas del modelo de procesos para métodos de medición.

III. TRABAJO RELACIONADO

Existen múltiples estudios que han analizado distintos aspectos sobre la automatización de la medición del tamaño funcional (FSM). El objetivo de estas investigaciones es automatizar el conteo del tamaño funcional a partir de distintos artefactos de *software* tales como la especificación de requerimientos, los modelos realizados durante el diseño, el código fuente de la aplicación o el análisis de aplicaciones de *software* en tiempo de ejecución [13]. En [14, 13] se realiza un mapeo de métodos y procedimientos de medición donde se identifican posibilidades de automatización y las principales limitantes de este tipo de estudios.

Ozkan & Demirors [15] analizan propuestas de formalización FSM, estas buscan resolver las ambigüedades de los métodos de medición existentes, sus conceptos y sus reglas de aplicación, e intentan reducir la variabilidad que genera la interpretación de las reglas generales de los métodos de medición. Determinan que la automatización de los procesos de medición se debe realizar a partir de la interpretación consistente de las reglas de medición, para garantizar la consistencia y la repetibilidad. Ozkan [16] propone una herramienta de medición de tamaño funcional para aplicaciones con arquitectura de tres capas basada en el método COSMIC.

Marin, Giachetti & Pastor [17] analizan propuestas de medición FSM que utilizan modelos conceptuales como artefactos de entrada. Determinan la necesidad de establecer claramente las fases de la estrategia de medición, las reglas de mapeo entre los conceptos del método y los artefactos de entrada, y las reglas de identificación de los elementos funcionales del método FSM. Determinan que la automatización es necesaria para reducir el costo del conteo e incrementar la eficiencia del proceso de medición. En [4] se presenta un marco de trabajo de referencia para profesionales con el conjunto esencial de funciones para la implementación de estándares FSM y se incluye una recopilación de herramientas de medición disponibles. Barkallah, Gherbi & Abran [18] proponen un marco de trabajo para la automatización de procedimientos FSM para COSMIC basados en modelos UML como artefactos de entrada. Determinan que existe poca evidencia sobre la validación de las herramientas existentes que proponen la automatización del conteo.

Akca & Tarhan [19, 20] proponen una librería de medición en tiempo de ejecución basada en el método COSMIC para aplicaciones de negocios desarrolladas en Java. En [21, 22], Tarhan presenta una herramienta para medir las aplicaciones mediante el rastreo de los procesos funcionales y en [23] propone un conjunto de requerimientos que deben ser considerados por las herramientas de medición que utilizan el código fuente como artefacto de entrada. Lamma, Mello, Riguzzi [24] presentan una herramienta de medición para especificaciones basadas en diagramas de entidad relación y de flujos de datos. En [25, 26] los autores realizan la medición a partir del mapeo de elementos UML hacia los componentes del método de medición. Abrahao y Pastor [27] realizan el proceso de medición sobre modelos conceptuales y en [28] proponen un procedimiento de medición para calcular aplicaciones orientadas a objetos de acuerdo al método IFPUG FPA.

En Edagawa et al [1] aplican un proceso automatizado de conteo a partir del análisis de las transacciones de la interfaz y los accesos a datos. Finalmente, el *Object Management Group* provee una especificación para la automatización del conteo de puntos de función a partir del código fuente de un sistema [29]. Esta especificación es usada para medir sistemas transaccionales y el proceso

difiere del método IFPUG FPA donde los criterios subjetivos deben ser reemplazados por las reglas requeridas para la automatización. Para nuestro estudio, se implementa una herramienta basada en los trabajos descritos en [1, 24, 23, 29] y las experiencias adquiridas en estudios empíricos previos realizados sobre automatización [30, 31, 32].

IV. Herramienta de medición

Esta sección detalla la herramienta prototipo de medición que incluye el detalle del marco de trabajo de desarrollo de la organización, la descripción del módulo de medición y la explicación del proceso de medición automatizado del tamaño funcional.

A. Marco de trabajo de desarrollo (FastWorks)

La organización de desarrollo de *software* Grupo Asesor en Informática S.A, en la cual se desarrolla este estudio, cuenta con un marco de trabajo para el desarrollo de aplicaciones de *software* llamado FastWorks (FW). Este consiste en una herramienta de especificación y generación de código que permite el modelado y desarrollo de aplicaciones transaccionales suministrando una arquitectura estandarizada que soporta la implementación de los módulos y sus funcionalidades. FW ha sido desarrollado y mantenido por la organización internamente durante más de 15 años, obteniendo gran cantidad de casos de éxito en el desarrollo de aplicaciones para sus clientes, en su mayoría en el sector de Gobierno. FW se ha utilizado para el desarrollo de sistemas de información basados en la plataforma Microsoft .NET.

El objetivo del marco de trabajo es reducir el tiempo de desarrollo y mejorar la calidad de las aplicaciones que se producen. Los principales beneficios del uso de FW son: la estandarización del código fuente, mejorando la calidad y la mantenibilidad de las aplicaciones y el incremento en la productividad del proceso de desarrollo. Este marco de trabajo genera aplicaciones con una arquitectura genérica y estandarizada que abstrae la complejidad técnica y permite a los equipos de desarrollo enfocarse en las necesidades del negocio y no en los aspectos técnicos ya resueltos por el marco de trabajo. Una de las principales ventajas competitivas del uso de FW es obtener un código fuente, arquitectura e interfaz de usuario estándar y con gran cantidad de componentes reutilizables.

El marco de trabajo FW provee un ambiente gráfico que soporta el modelado de las aplicaciones transaccionales. El proceso de modelaje es realizado en el componente llamado “especificador”. Este componente provee un conjunto de áreas de trabajo para realizar la especificación del modelo. A partir del modelo generado, se utiliza el componente llamado “generador” para obtener el código fuente de la aplicación. En este trabajo, se implementa el módulo del marco de trabajo que automáticamente genera el modelo funcional de la aplicación y realiza la medición de tamaño funcional. En el proceso de producción de *software* del marco de trabajo se modelan los siguientes elementos y sus relaciones para representar una aplicación de *software*:

- Módulo (*module*): una aplicación se compone de uno o más módulos. Un módulo se compone de una o más ventanas y donde se define una ventana principal para cada módulo.
- Ventana (*window*): es una pantalla en el módulo que puede contener uno o más marcos, con interacción entre ellos y una organización específica en la ventana.
- Marco (*frame*): contiene una o más sentencias que permiten la administración de datos en el marco, uno o más controles, acciones y eventos. Los marcos se clasifican en tipo Listado (LL) que contienen un único control de tipo tabla o tipo Formulario (FF) y contiene múltiples controles.
- Base de datos (*database*): esta detalla las entidades de datos de la aplicación y todos sus atributos.
- Sentencia (*statements*): corresponde a una sentencia de programación para la administración de los datos de un marco. Esta puede incluir sentencias SQL para las bases de datos, además pueden representar llamadas a vistas, procedimientos y funciones almacenadas. Existen sentencias de marco y de acción.
- Operación (*operation*): corresponde a una operación de programación para la administración de los datos que se realiza en respuesta a una solicitud de datos o un evento realizado por el usuario.
- Control (*control*): permite a un usuario la visualización, entrada o salida de los datos en un marco. Un control puede contener otros controles y puede tener relacionados uno o varios eventos.
- Acción (*action*): permite la ejecución de transacciones en el marco al usuario basado en uno o más sentencias. Una transacción permite realizar operaciones sobre otros controles o sobre las bases de datos.
- Evento (*event*): permite la ejecución de transacciones como respuesta a eventos realizados por el usuario en el marco. Puede realizar operaciones sobre otros controles o sobre la base de datos mediante una acción.
- Jerarquías (*hierarchy*): permiten organizar las ventanas posibilitando ventanas padre y una o varias ventanas hijas.
- Contexto (*context*): representa la información global de la aplicación. Habilita un diccionario disponible para que las ventanas, marcos y controles puedan leer o escribir valores para establecer relaciones o interacciones entre ellos.

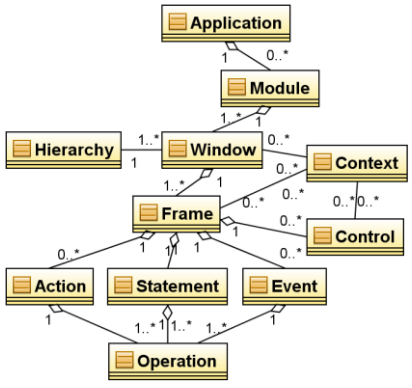

Figura 1 Metamodelo del marco de trabajo.

La Figura 1 detalla el metamodelo para la construcción de aplicaciones bajo el marco de trabajo. En las primeras fases del proceso de desarrollo con FW se realiza el modelo de la aplicación que incluye los módulos, la base de datos y todos los elementos descritos anteriormente. El modelo del marco de trabajo describe todas las funcionalidades del usuario final. A partir del modelo, se genera una aplicación completamente funcional. En nuestro caso, el modelo de la aplicación contiene toda la información necesaria para la construcción del modelo funcional utilizado para obtener el tamaño funcional.

La Figura 2 muestra el componente "especificador" del marco de trabajo FW. Este cuenta con un conjunto de áreas de trabajo para realizar la especificación del modelo que se describen a continuación:

- Área de estructura: permite la visualización de los diferentes componentes que conformen un elemento de especificación.
- Área de propiedades: permite la edición de las propiedades del elemento de especificación o de los componentes que lo conformen. Estas propiedades pueden ser de un componente seleccionado del área de estructura o del área visual.
- Área visual: permite la edición de los diferentes componentes visuales que conformen un elemento de especificación.

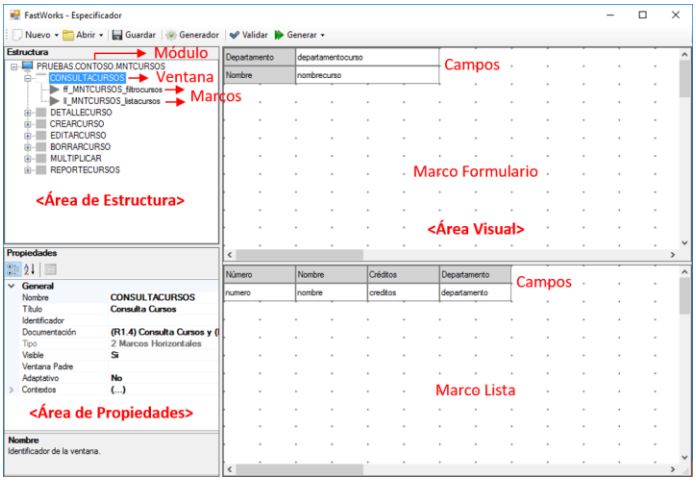

Figura 2 Herramienta de especificación de marco de trabajo.

B. *Módulo de medición (FastWorks FPA)*

El módulo de medición, que fue implementada como parte del marco de trabajo de desarrollo, construye un modelo funcional a partir del modelo de la aplicación que es utilizado como entrada para aplicar el procedimiento de medición y obtiene automáticamente el tamaño funcional. La herramienta fue desarrollada en la plataforma .NET utilizando las librerías MS ScriptDom para el análisis de transacciones funcionales y el API de Neo4j para la administración del grafo de representación del modelo funcional de la aplicación y el cálculo del tamaño funcional. El modulo implementa los siguientes componentes, tal como se muestra en la Figura 3:

- Generador: este componente genera un archivo de interfaz XML a partir del modelo de la aplicación que detalla toda la información requerida para la construcción del modelo funcional. La generación del archivo de interfaz se genera a partir de la instanciación de los objetos del modelo e implementa un conjunto de características necesarias para la automatización tales como: identificación de los eventos de usuario desde la interfaz de usuario de la aplicación, de procesos funcionales que representan la funcionalidad del usuario, de las entidades de datos en almacenamiento persistente y en interfaces hacia otras aplicaciones y de los tipos de acceso de las transacciones realidades desde los eventos hacia las entidades de datos.
- Constructor: este componente procesa el archivo de interfaz XML y construye el modelo funcional que incluye el modelo de eventos, de datos y de transacciones. Este componente simula la ejecución de transacciones y genera un grafo de representación funcional de la aplicación.
- Medidor: este componente (a) permite realizar el mapeo entre los elementos del modelo funcional y los del método IFPUG FPA, (b) identifica los procesos funcionales únicos e independientes que incluyen las transacciones y los datos y (c) calcula el tamaño funcional de acuerdo a la aplicación de las reglas de mapeo y de asignación numérica para cada uno de los procesos funcionales y almacena los resultados del conteo de tamaño funcional. Además, provee la posibilidad de ver y editar los resultados del mapeo y recalcular los resultados de medición.
- Reportador: este componente presenta el detalle de los resultados del procedimiento de medición. Este detalla todas las etapas del proceso y desglosa cada una de las mediciones parciales. Además, permite mantener la trazabilidad entre los resultados de la medición, el modelo funcional y el modelo de la aplicación. Finalmente, presenta una visualización del modelo funcional. A partir del resultado que proporciona este componente se realiza el proceso de aplicación del protocolo de verificación de exactitud.

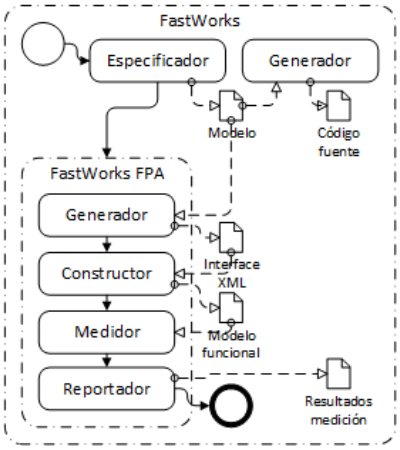

Figura 3 Herramienta de medición.

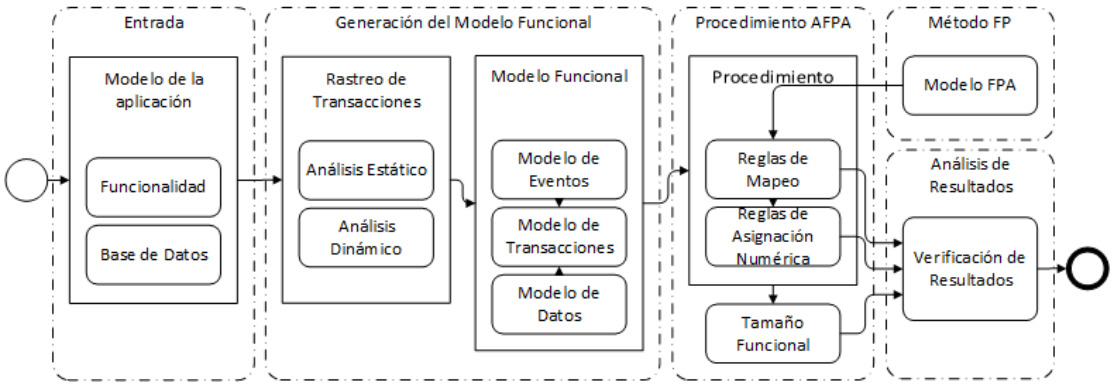

Figura 4 Procedimiento de medición automatizado.

La Figura 4 detalla el proceso para obtener el tamaño funcional aplicando el procedimiento de medición. Este recibe como entrada el modelo definido bajo el marco de trabajo de desarrollo. Primero, se construye el modelo funcional, a partir del análisis del modelo de la aplicación y cada uno de sus elementos. Este se obtiene mediante el análisis estático y dinámico del modelo. En el proceso de análisis se determinan los eventos del usuario final y se rastrean las transacciones hacia las entidades de datos. Segundo, el tamaño funcional se obtiene a partir del modelo funcional construido aplicando las reglas de mapeo de elementos y asignación numérica para obtener el tamaño funcional basado el método IFPUG FPA. La etapa de mapeo entre elementos de la aplicación y del método de medición permite mantener la trazabilidad entre los resultados de medición, los accesos y movimientos de datos, los elementos del modelo de la aplicación y sus requerimientos funcionales. Finalmente, se realiza la verificación de los resultados de medición.

1) Definición de los metamodelos

Un metamodelo detalla la estructura de representación de los componentes de una aplicación de *software*. Este define los elementos básicos y las relaciones a partir de las cuales se determinan las reglas de identificación, mapeo y medición para obtener el tamaño funcional. En nuestro procedimiento se utilizan tres metamodelos: (1) el metamodelo del marco de trabajo de la organización (Figura 1), (2) el metamodelo funcional definido por el procedimiento (Figura 5) y (3) el metamodelo del método IFPUG FPA (Figura 6). Durante la aplicación del procedimiento de medición, primero realizamos el mapeo entre el metamodelo del marco de trabajo y el metamodelo del modelo funcional. Segundo, realizamos el mapo entre el metamodelo del modelo funcional y el metamodelo del método de medición IFPUG FPA. Tercero, los componentes del modelo funcional identificados son mapeados a valores numéricos mediante las reglas de medición que obtienen el tamaño de la funcionalidad de la aplicación.

La Figura 5 detalla el metamodelo de representación funcional requerido para realizar el conteo de puntos de función. Este detalla la información necesaria que se debe recolectar desde la aplicación para construir el modelo funcional que permiten la aplicación de las reglas de mapeo y asignación numérica. Este se construye a partir del análisis de los eventos y transacciones obtenidas durante la simulación de la ejecución de los procesos funcionales. En nuestro caso, el modelo de la aplicación delimita las fronteras de la aplicación a ser medida. El modelo funcional requiere la identificación de los siguientes elementos:

- Aplicación (*application*): representa la aplicación a ser medida. Los artefactos de entrada determinan los límites de la medición. Una aplicación se puede componer de uno o muchos controles.
- Control (*control*): es un componente de interfaz de usuario y puede contener otros controles. El usuario final accede las funcionalidades cuando ejecuta los eventos asociados a los controles. Un control puede tener asociados uno o varios eventos. Ejemplos: ventanas, botones, campos de edición, tablas de datos, entre otros.
- Evento (*event*): permite la ejecución de una opción del usuario. Un evento puede ser clasificado como tipo función o navegación. La navegación permite ir a otro control de interfaz de usuario sin realizar ninguna transacción de datos. La función realiza accesos a las entidades de datos por medio de transacciones de lectura o escritura. Un evento puede ejecutar ninguna o muchas funciones transaccionales y puede ser clasificado como un evento de navegación o de función.
- Función (*function*): representa un proceso funcional. Puede ser clasificada como una transacción (lectura, escritura [insertar, actualizar, borrar]) o una entidad de datos. Ambos tipos de función deben ser identificadas en el proceso de creación del modelo funcional. Una transacción de datos accede una entidad de datos de la aplicación mediante la ejecución de una funcionalidad del usuario.
- Transacción (*transaction*): es una secuencia de operaciones hacia la base de datos realizados como una sola unidad (desde que el usuario dispara un evento desde la interfaz hasta que la aplicación devuelve el control al usuario). Una transacción de datos accede una o varias entidades de datos de la aplicación por medio de operaciones de lectura (*read*) o escritura (*write*).
- Entidad de datos (*data entity*): en un archivo o tabla de la base de datos o un grupo de archivos o tablas que representan un agrupamiento lógico. Un agrupamiento lógico son todas las entidades de datos que se relacionan entre si y que son vistas

como una sola unidad desde la perspectiva del usuario final. Cada entidad de datos se compone de elementos de datos.

- Elemento de datos (*data element*): es un atributo o columna de un archivo o tabla de base de datos que puede representar un campo físico o calculado.
- Base de datos (*database*): representa las bases de datos con las cuales la aplicación interactúa durante la ejecución de las funcionalidades del usuario.

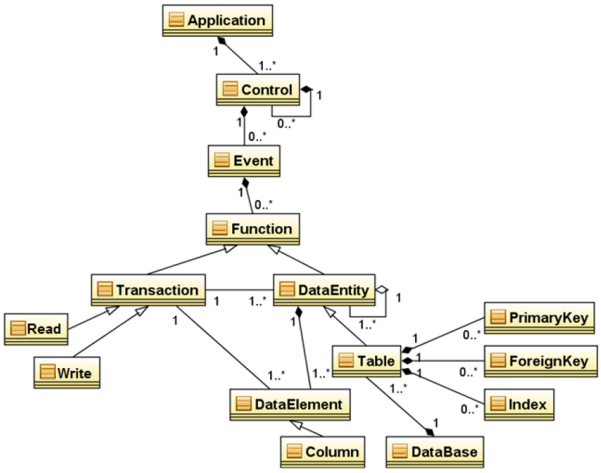

Figura 5 Metamodelo del procedimiento de medición.

2) Generación de los modelos

Basado en el modelo de la aplicación desarrollado en el marco de trabajo, el componente "Generador" extrae los elementos requeridos por el procedimiento de medición y realiza el proceso de generación automática del archivo de interface XML que permite la construcción del modelo funcional. La generación de la interface implica la transformación de los elementos entre los metamodelos. La generación del archivo de interface conlleva los siguientes pasos:

- P1. Se seleccionan los módulos de la aplicación a medir y que formarán parte en el proceso de generación.
- P2. Se realiza la instanciación y lectura de todos los módulos seleccionados. Esto produce para cada uno de los módulos una instancia del modelo de objetos que representa cada uno de los módulos.
- P3. Para cada módulo se recorren cada uno de los elementos del modelo (ventanas, marcos, controles, sentencias, acciones, eventos y operaciones) y se aplican las reglas de mapeo entre los metamodelos.
- P4. Se genera el archivo de interface XML para cada uno de los módulos.

Las reglas de mapeo entre el metamodelo del marco de trabajo y del modelo funcional son las siguientes:

- R1. Un módulo genera al elemento aplicación. Los límites de la aplicación se determinan por el modelo que representa un módulo de la aplicación.
- R2. Cada ventana, marco y control genera un control. Para todo control, si el control tiene una sentencia relacionada se define un evento.
- R3. Cada sentencia, acción y evento se genera un evento. En este evento la sentencia de programación, acción y evento se trata como una función
- R4. Cada operación se genera una función con sus respectivas transacciones. Cada sentencia de programación de base de datos se mapea a una función del evento mediante una o más transacciones.
- R5. La base de datos genera las entidades y elementos de datos y un elemento de base de datos.

3) Construcción del modelo funcional

Con el archivo de interface realizamos la construcción del modelo funcional de la aplicación. La construcción del modelo funcional se realiza siguiendo los tres pasos que se describen a continuación.

- P1. Modelo de eventos. Se identifican todos los controles que representan la interfaz de usuario final. Para cada control, se identifican todos los controles asociados que cuenten con eventos de usuario final. Para cada evento se identifican y registran todas las entradas requeridas para la ejecución del evento. Si el evento dispara una transacción de la base de datos, se registra la transacción como un evento funcional. Si no, se registra la transacción como un evento de navegación. Para cada evento se registra el control de interfaz y el control específico que generó el evento, el nombre del evento y la lista de funciones transaccionales que realizó en la base de datos. El modelo de eventos representa todas las funcionalidades que el usuario puede realizar desde la interfaz de usuario.
- P2. Modelo de datos. Se identifican todas las entidades de datos (almacenamiento persistente) para las bases de datos y otras estructuras de datos que accede la aplicación. Para cada entidad de datos, se identifican los elementos de datos (atributos) relacionados, los identificadores únicos y las relaciones entre grupos de datos. El modelo de datos asocia cada entidad con sus elementos de datos y relaciona las entidades de datos entre sí.
- P3. Modelo de transacciones. Se identifican las funciones y accesos que realiza cada uno de los eventos en las entidades y los elementos de datos. Cada uno de los accesos a los datos se marca como una transacción de lectura o escritura. Si un evento realiza múltiples accesos a un elemento de datos se administran los duplicados para no afectar el conteo del tamaño funcional. El modelo de transacciones registra todas las transacciones realizadas sobre cada elemento de datos desde la interfaz de usuario.

La unión de los modelos de eventos, datos y transacciones constituye el modelo funcional. Este se representa mediante una estructura de grafo dirigido, etiquetado y ponderado donde sus nodos representan cada uno de los elementos funcionales y las aristas representan las relaciones entre cada uno de ellos donde V es un conjunto finito de elementos no vacío y $E \subseteq V \times V$. El par (V, E)

es un grafo dirigido sobre V, donde V es el conjunto de nodos (vértices) y E es su conjunto de aristas, tal como se muestra en la Figura 6. Escribimos $G = (V, E)$ para denotar el grafo. En nuestro caso el modelo funcional G es un conjunto de elementos N y sus relaciones E, donde $V \in \mathrm{T}_v$ y $\mathrm{T}_v = \{$*Application*, *Control*, *Event*, *Function*, *Read*, *Write*, *Table*, *Column*, *PrimaryKey*, *ForeignKey*, *Index*, *Database*$\}$. Además, $E \in \mathrm{T}_e$ donde $\mathrm{T}_e = \{$*Father*, *Select*, *Insert*, *Update*, *Delete*, *Relationship*$\}$.

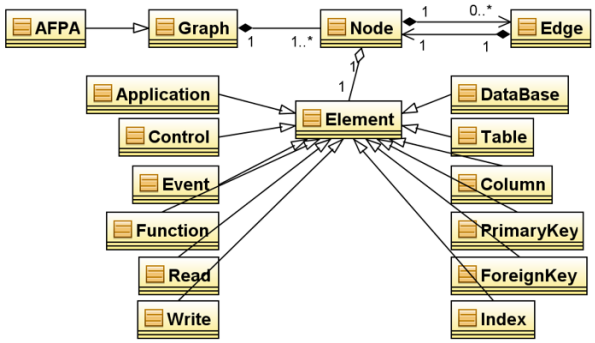

Figura 6 Grafo de representación del modelo funcional.

4) Mapeo entre modelos

Este modelo funcional es mapeado al metamodelo IFPUG FPA. El mapeo de los elementos del modelo funcional hacia el metamodelo IFPUG FPA se realiza sobre el subgrafo que representa las transacciones funcionales. El grafo completo se utiliza para mantener la trazabilidad del conteo funcional hacia la interfaz de la aplicación que representa los requerimientos del usuario. La Figura 7 detalla el metamodelo del método IFPUG FPA que define la información que debe ser representada por el modelo funcional. Una aplicación bajo medición representada mediante el método IFPUG FPA se compone de los componentes funcionales descritos en la Sección II.

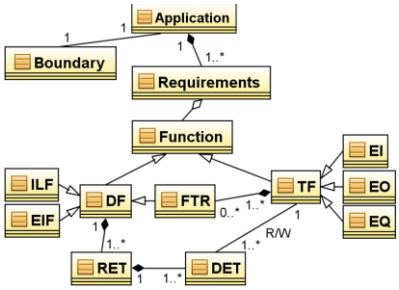

Figura 7 Metamodelo IFPUG FPA.

En este caso, el procedimiento de medición se encarga de instanciar el metamodelo IFPUG FPA automáticamente a partir del modelo funcional.

La Figura 8 presenta el mapeo entre los metamodelos. Las reglas de mapeo establecen las relaciones entre los elementos de los metamodelos para identificar los componentes del método de medición. Con base en el modelo funcional se determinan los candidatos a funciones transaccionales (TF), a archivos referenciados (FTR) y a elementos de datos referenciados (DET). Posteriormente, se clasifican los candidatos TF a candidatos a entradas (EI), salidas (EO), o consultas (EQ) de acuerdo al tipo de acceso realizado sobre los FTRs y considerando si utiliza elementos calculados con base en los elementos de datos. De acuerdo a los accesos simultáneos realizados en las entidades de datos y las relaciones entre las entidades de datos identificadas en el modelo de datos, se determinan los agrupamientos lógicos de entidades para clasificar las funciones de datos (DF) candidatas y sus respectivos RET asociados. Finalmente, de acuerdo al tipo de acceso realizado sobre los DFs y sus RETs, las entidades de datos se clasifican como candidatos a archivos internos (ILF) o externos (EIF). Durante el proceso de mapeo, las relaciones entre los conceptos de ambos metamodelos determinan los candidatos a considerar como conceptos del método IFPUG FPA.

En el procedimiento propuesto, un elemento *function* se relaciona con el concepto función transaccional (TF), un elemento *read* con una salida (EO) o una consulta (EQ), el *write* con una entrada (EI), un elemento *table* con un archivo lógico interno (ILF) o una interface externa (EIF), o con una función de datos (DF), elemento de tipo registro (RET) o elemento de tipo archivo referenciado (FTR). Finalmente, un elemento c*olumn* se relaciona con el concepto de elemento de datos referenciado (DET). A partir del modelo funcional del procedimiento se determinan los candidatos para cada uno de los componentes funcionales del método de medición IFPUG FPA. Los candidatos son analizados por el procedimiento de medición para determinar si cumplen las características de un elemento funcional. El mapeo de los componentes entre el metamodelo del procedimiento y del método IFPUG FPA se realiza a partir de la aplicación de las siguientes reglas:

- R1. Para cada elemento tipo *event* se mapea un candidato a función transaccional (TF) si tiene asociado uno o varios elementos tipo *function*. Si no es un evento de navegación.
- R2. Para cada elemento tipo *function* identificamos los elementos tipo *column* asociados el cual se mapea como un elemento de datos (DET). Identificamos los elementos tipo *table* asociados el cual se mapea como un archivo referenciado (FTR).
- R3. Para cada elemento tipo *function* identificamos los elementos tipo *read* o *write* asociados y para cada uno de ellos los elementos tipo *column*. Si existe al menos una relación con un elemento *write* se mapea como una operación de entrada de datos (EI). Si existe solo una relación con elementos *read* e identificamos solo elementos *column* tipo *field* se mapea como una operación de consulta (EQ), si no se mapea como una operación de salida de datos (EO).
- R4. Para cada elemento tipo *read* o *write* identificamos los elementos tipo *table* asociados. Si los elementos tipo *table* asociados son mayor que uno entonces identificamos si pertenecen a un mismo grupo lógico de la siguiente manera: (a) Las asociaciones maestro-detalle de grupos lógicos se determinan por los elementos relacionados por una llave foránea que forma parte de una llave primaria de la tabla. (b) Las relaciones de asociación en grupos lógicos se determinan por

múltiples llaves primarias que son llave foránea de todas las columnas de otra tabla. Si existe una asociación entre tablas, estas pertenecen a un agrupamiento lógico y se mapean como un único grupo lógico (FTR). Si solo existe un elemento tipo *table* asociado entonces identificamos si no pertenece a un grupo lógico existente, si es así entonces se mapea como un nuevo grupo lógico (FTR).

- R5. Cada grupo lógico único se mapea como una función de datos (DF) de la siguiente manera: se mapea como un archivo interno (ILF) si al menos tiene una asociación con un elemento *write* y como una interfaz externa (EIF) si solo se asocia con elementos *read.* Todos los elementos tipo *table* que no se encuentran asociados a un elemento tipo *function* no se consideran parte de la funcionalidad y son marcados como tablas técnicas.

A partir de la identificación de los componentes funcionales, en la fase de medición se calcula el tamaño funcional en puntos de función.

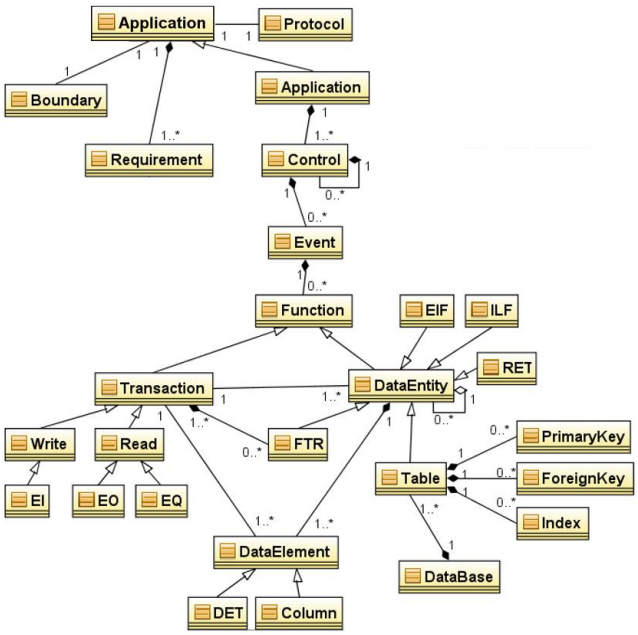

Figura 8 Mapeo entre metamodelos.

5) Medición del modelo funcional

Las reglas de asignación numérica se aplican de acuerdo al método de medición IFPUG FPA para obtener los resultados del tamaño funcional. En la fase de mapeo de conceptos entre metamodelos se identifican los componentes funcionales a partir de la representación del grafo del modelo funcional. En la fase de medición se cuantifica el tamaño funcional en puntos de función sin ajustar (UFP) mediante la aplicación de las siguientes reglas:

- R1. Cada elemento tipo *function* se cuenta como función transaccional (TF). Si existe una relación con un elemento *write* se cuenta como entrada de datos (EI). Si existe una relación con un elemento *read* y solo elementos *column* tipo *field* se cuenta como consulta (EQ), si no se cuenta como salida de datos (EO).
- R2. Cada elemento tipo *column* asociado a una *function* se cuenta como un elemento de datos (DET) para un grupo lógico eliminando duplicados, esto se realiza restando los elementos llave foránea. Al total de DET se le suman dos adicionales por la ejecución de la acción y control de mensajes hacia el usuario final. Si un grupo lógico está compuesto por más de un elemento *table* el total RET es la cantidad de elementos *table* menos uno, si no la cantidad RET es uno.
- R3. Cada grupo lógico asociado a una *function* se cuenta como un archivo referenciado (FTR). Cada grupo lógico se cuenta como una función de datos (DF). Si al menos tiene una asociación con un elemento *write* se cuenta como archivo interno (ILF) y como una interfaz externa (EIF) si se asocia solo con elementos *read*.

A partir de los conteos aplicamos el procedimiento del método IFPUG FPA para calcular el de puntos de función sin ajustar (UFP) final.

C. Protocolo de verificación

El protocolo de verificación de exactitud permite validar y calibrar el procedimiento de medición propuesto. Este propone una evaluación *top-down* de los resultados de medición para identificar diferencias durante la aplicación del proceso de medición con los valores esperados en comparación con un *true value* obtenido a partir de la medición de un experto. A partir de los resultados del proceso de medición se verifica cada uno de los componentes funcionales y su tamaño a partir del protocolo propuesto en [31]: (F1) Comparación de los resultados totales de medición, (F2) Comparación detallada de la exactitud de los resultados y (F3) Identificación y recuperación de errores.

V. VALIDACIÓN

A. Diseño

Para validar los resultados de la herramienta, aplicamos el protocolo de verificación a los resultados obtenidos sobre una aplicación modelada con el marco de trabajo. Los resultados son comparados con los obtenidos durante la medición manual de los requerimientos de la aplicación. En nuestro caso de medición investigamos las siguientes preguntas de investigación: (RQ1) ¿Cuál es la exactitud del total de puntos de función obtenido por el procedimiento en comparación con la medición realizada sobre la especificación de requerimientos? (RQ2) ¿Cuál es la exactitud de los puntos de función obtenidos por el procedimiento para cada uno de los componentes funcionales básicos (BFC) en comparación con la medición realizada sobre la especificación de requerimientos?

La aplicación medida fue modelada de manera independiente por uno de los profesionales de la organización basado en el documento de requerimientos. Los requerimientos fueron descritos en términos de funcionalidad utilizando el estándar IEE 830. La aplicación es un sistema transaccional con funcionalidades para la administración de la matrícula de una organización educativa. Esta

principalmente ofrece operaciones CRUDL (*create, read, update, delete, list*) con las siguientes funciones: mantenimiento de cursos (crear, editar, borrar, reporte, búsqueda, asignación de departamento), mantenimiento de instructores (crear, editar, borrar, reporte, búsqueda, asignación de cursos, reporte), mantenimiento de departamentos (crear, editar, borrar, reporte, asignación de administrador), y mantenimiento de estudiantes (crear, editar, borrar, reporte, búsqueda).

Las limitaciones se relacionan con las amenazas a su validez. Dos investigadores aplicaron el método IFPUG FPA para contar el tamaño funcional utilizado como el *true value*. Aunque tienen experiencia en FSM, no son contadores certificados. Dado que se ha aplicado el procedimiento y herramienta de medición a una aplicación transaccional, el nivel de exactitud depende directamente de las características de la aplicación bajo medición. La aplicación medida no cuenta con interfaces externas (EIF), por tanto con aplicaciones que cuenten con múltiples EIF se debe evaluar si las reglas de lectura y escritura sobre los archivos de la aplicación modifican los niveles de exactitud de los resultados. Del mismo modo, en la identificación de las salidas (EO) y consultas (EQ) el procedimiento considera solo los campos calculados programados en la capa de base de datos lo que puede afectar la exactitud en la clasificación de los componentes en aplicaciones con diferentes características. Finalmente, las decisiones sobre la implementación del modelo de base de datos pueden influenciar la identificación los grupos lógicos basado en la reglas del procedimiento. Los resultados obtenidos pueden ser generalizados al ambiente de desarrollo delimitado por el marco de trabajo y el modelo descrito en este estudio.

B. Resultados

La aplicación medida consta de 32 requerimientos principalmente relacionados con operaciones CRUDL. Al realizar la comparación entre los procesos de medición obtuvimos que en el conteo manual se midieron 164 UFP y en el conteo automatizado 155 UPF. Con respecto a la RQ1, los resultados presentan un nivel de exactitud aceptable (95%) pero no exacto. Considerando el porcentaje de variabilidad observado en estudios previos y la variabilidad considerada aceptable para mediciones de expertos, de un máximo del 10% entre dos conteos, los resultados se pueden considerar prácticos para la industria.

Con respecto a la RQ2, para el conteo manual se extraen 32 funciones transaccionales (TF), que se componen de 15 entradas (EI), 4 salidas (EO) y 13 consultas (EQ). Además, se extraen 5 funciones de datos (DF) que a su vez se clasifican como archivos internos (ILF) y no se identifican interfaces externas (EIF). Las TF corresponden a 129 puntos de función sin ajustar (UFP) y las DF a 35 UFP. En el conteo automatizado se extraen 30 TF, que se componen de 15 EI, 2 EO y 13 EQ. Se extraen 5 DF que se clasifican como ILF. De igual manera, no se identifican EIF. Las TF corresponden a 120 UFP y las DF a 35 UFP. Al analizar los conteos para las funciones transaccionales (TF) y las funciones de datos (DF) se obtienen resultados que se pueden considerar aceptables. Los conteos de cada uno de los BFC presentan variaciones de 5.2% para las EI. Para los EQ es del 0.0%. En el caso de las EO, la variación es del 57.1% las cuales representan resultados no aceptables en la industria. Los resultados muestran que el total de puntos de función (UFP) y los componentes funcionales básicos (BFC) EI, EQ y ILF podrían ser medidos con un porcentaje de exactitud aceptable; sin embargo, estos resultados no se obtuvieron para el componente EO.

Al realizar el análisis de los resultados de medición aplicando el protocolo de verificación se obtienen las causas principales (factores de influencia) de las diferencias encontradas. Las diferencias se producen por diferentes clasificaciones para dos requerimientos cuyos componentes EO fueron clasificados como EQ. Por otro lado, existen diferencias por la identificación de los elementos DET, RET y FTR. En algunos casos las variaciones presentadas se dan por implementación técnica del marco de trabajo. Finalmente, en el modelo de la aplicación es posible detallar opciones de filtrado de manera automática bajo una misma implementación lo hizo que varios requerimientos documentados se unificaran en uno solo en el modelo reduciendo el valor del conteo. En el proceso de calibración de procedimiento se identifican algunas tablas y columnas técnicas que no forman parte de los requerimientos del usuario, componentes de la interfaz que realizan funcionalidad adicional para carga y uso de la información de las bases de datos que no se contempla en el conteo y eventos adicionales que obtienen datos y repiten funcionalidad realizada en otros requerimientos. El análisis del modelo funcional representado por los grafos permite la identificación de estos elementos para calibrar los resultados. Para nuestro caso de estudio, estos casos se consideraron como funcionalidad duplicada pero de igual manera se reporta al profesional para ser considerada en sus procesos de toma de decisiones.

VI. DISCUSIÓN

Las principales lecciones aprendidas son las siguientes:

- El modelo del procedimiento de medición es simple y puede representar de manera completa los elementos funcionales de una aplicación.
- Con la información extraída de los modelos del marco de trabajo es posible realizar la medición de la aplicación. Además, da la oportunidad a partir de su visualización, obtener información importante para el análisis de las aplicaciones.
- El procedimiento de medición debe ser calibrado de acuerdo a los estándares de modelado del marco de trabajo de la organización y sus características únicas.

Por otro lado, las principales limitaciones de la herramienta de medición son las siguientes:

- Aunque el procedimiento puede ser adaptado para otros entornos de desarrollo, en su estado actual, solo se puede aplicar en aplicaciones desarrolladas bajo el modelo del marco de trabajo.
- Las decisiones tomadas para la calibración de la herramienta pueden afectar los conteos si el modelado de las aplicaciones no siguen los estándares de desarrollo definidos bajo el marco de trabajo y el procedimiento de medición.
- El procedimiento de medición solo obtiene el tamaño en puntos de función sin ajustar (UFP) del método IFPUG FPA.

- Esto es un trabajo e progreso, es requerido chequear y calibrar las reglas de clasificación de componentes EO, EQ.

VII. Conclusiones y trabajo futuro

En este estudio se presentó un procedimiento FSM basado en el método IFPUG FPA para obtener el tamaño funcional a partir de un modelo de *software* expresado en el marco de trabajo FastWorks (FW). Se implementó una herramienta de medición prototipo y se realizó un caso de estudio para validar la exactitud de los resultados de medición. Los resultados obtenidos ofrecen evidencia de que se puede obtener mediciones de tamaño funcional con un nivel aceptable de exactitud que puede permitir a la organización la implementación de un sistema de métricas basado en el tamaño funcional de sus aplicaciones.

Como trabajo futuro la organización aplicará el procedimiento de medición a las aplicaciones de su portafolio de proyectos y validará los resultados de medición para incorporar la métrica de tamaño funcional en su programa de métricas y los investigadores calibrarán el procedimiento de medición y la herramienta de medición a partir de los resultados obtenidos en los múltiples casos de estudio. Además, se realizarán estudios para analizar la posibilidad de obtener el conteo de puntos de función COSMIC FFP a partir del uso de los modelos definidos por el procedimiento de medición.

Agradecimientos

Este estudio fue apoyado por el Ministerio de Ciencia, Tecnología y Telecomunicaciones (MICITT) y la Universidad de Costa Rica Proyecto No. 834-B5-A18. Agradecemos a Grupo Asesor en Informática S.A. por su valioso aporte en este trabajo.

Referencias

[1] Edagawa, T., Akaike, T., Higo, Y., Kusumoto, S., Hanabusa, S., Shibamoto, T. “Function point measurement from Web application source code based on screen transitions and database accesses”. Journal of Systems and Software, 84(6), 976-984, 2011.

[2] Garmus, D., Herron, D. “Function point analysis: measurement practices for successful software projects”. Addison-Wesley Longman Publishing Co., Inc., 2001.

[3] Fingerman, S. “Practical software project estimation; a toolkit for estimating software development effort & duration”. Sci-Tech News (Vol. 65). McGraw Hill Professional, 2011.

[4] Stambollian, A., Abran, A. “Survey of Automation Tools Supporting COSMIC-FFP ISO 19761”. En IWSM Mensura – MetriKon, 2006.

[5] Lavazza, L. “Automated function points: Critical evaluation and discussion”. En WETSoM. Vol. 2015-Augus, pp. 35-43, 2015.

[6] Bajwa, S., Gencel, C., Abrahamsson, P. “Software Product Size Measurement Methods: A Systematic Mapping Study”. En IWSM-MENSURA. 176-190, 2014.

[7] Soubra, H., Abran, A., Ramdane-Cherif, A. “Verifying the accuracy of automation tools for the measurement of software with COSMIC-ISO 19761”. En IWSM-MENSURA, pp. 23-31, 2014.

[8] Heller, R. “Automated Function Point Counting: A Fact Based Analysis”. Q/P Management Group, Inc. http://www.qpmg.com/Library/confirm _ down-load.php?id=49&pid=1.

[9] ISO. “ISO/IEC 20926, Software and systems engineering - Software measurement – IFPUG functional size measurement method”, 2009.

[10] Albrecht, A. “Measuring application development productivity”. En SHARE/GUIDE/IBM, Vol. 10, pp. 83-92, 1979.

[11] Jones, C. “Function points as a universal software metric”. ACM SIGSOFT Software Engineering Notes, 38(4), 1, 2013.

[12] Abran, A. “Software Metrics and Software Metrology”. John Wiley & Sons, New Jersey, 2010. doi: 10.1002/9780470606834

[13] Quesada-López, C., Jenkins, M. "Procedimientos de medición del tamaño funcional: un mapeo sistemático de literatura". En CIBSE, 2017.

[14] Bajwa, S. S., Gencel, C., Abrahamsson, P. Software Product Size Measurement Methods: A Systematic Mapping Study. En IWSM-MENSURA. 176-190, 2014.

[15] Ozkan, B., & Demirors, O. “Formalization Studies in Functional Size Measurement”. Modern Software Engineering Concepts and Practices: Advanced Approaches, 242, 2010.

[16] Ozkan, B. “Automated Functional Size Measurement for Three-Tier Object Relational Mapping Architectures”. Journal of Software Engineering, (21), 311-338, 2011.

[17] Marin, B., Giachetti, G., Pastor, O. “Measurement of Functional Size in Conceptual Models: A Survey of Measurement Procedures Based on COSMIC” En. IWSM-MENSURA. 170-183, 2008.

[18] Barkallah, S., Gherbi, A., Abran, A. “COSMIC functional size measurement using UML models”. En Software Engineering, Business and Education. pp. 137-146. Springer, 2011.

[19] Akca, A., Tarhan, A. "Run-time measurement of cosmic functional size for java business applications: Initial results". In IWSM-MENSURA. pp. 226-231, 2012. IEEE.

[20] Akca, A., Tarhan, A. "Run-time measurement of COSMIC functional size for Java business applications: Is it worth the cost?" In IWSM-MENSURA. pp. 54-59, 2013. IEEE.

[21] Sag, M., Tarhan, A. "Measuring COSMIC software size from functional execution traces of Java business applications". In IWSM-MENSURA. pp. 272-281, 2014. IEEE.

[22] Gonultas, R., Tarhan, A. "Run-Time Calculation of COSMIC Functional Size via Automatic Installment of Measurement Code into Java Business Applications". In Euromicro Software Engineering and Advanced Applications (SEAA), pp. 112-118, 2015. IEEE.

[23] Tarhan, A., Ozkan, B., İçöz, G. “A Proposal on Requirements for COSMIC FSM Automation from Source Code”. En IWSM-MENSURA. pp. 195-200, 2016. IEEE. DOI: 10.1109/IWSM-Mensura.2016.038.

[24] Lamma, E., Mello, P., Riguzzi, F. “A system for measuring function points from an ER-DFD specification”. Computer Journal 47 (3), 358–372, 2004.

[25] Uemura, T., Kusumoto, S., Inoue, K. "Function point analysis for design specifications based on the unified modeling language". Journal of Software Maintenance and Evolution 13 (4), 223–243, 2001.

[26] Cantone, G., Pace, D., Calavaro, G. "Applying function point to unified modeling language: conversion model and pilot study". In Software Metrics Symposium, pp. 3280–3291, 2004.

[27] Abrahao, S., & Pastor, O. "Measuring the functional size of web applications". International Journal of Web Engineering and Technology, 1(1), 5–16, 2003.

[28] Abrahao, S. "On the functional size measurement of object-oriented conceptual schemas: design and evaluation issues". PhD thesis. Universidad Politecnica de Valencia, Spain. 2004.

[29] Object Management Group, Automated Function Points (AFP) Version 1.0, OMG Document Number: formal/2014-01-03, 2014.

[30] Quesada-López, C., Madrigal, D., Jenkins, M. "An Empirical Evaluation of Automated Function Points". En CIBSE, pp. 151-165, 2016.

[31] Quesada-López, C., Jenkins, M. “Applying a Verification Protocol to Evaluate the Accuracy of Functional Size Measurement Procedures: An Empirical Approach”. In PROFES, pp. 243-250. Springer, 2015.

[32] Quesada-López, C., Jenkins, M. “An evaluation of functional size measurement methods”. En CIBSE, pp. 151-165, 2015.

[33] Habra, N., Abran, A., Lopez, M., Sellami, A. “A framework for the design and verification of software measurement methods”. Journal of Systems and Software, 81(5), 633-648, 2008. doi: 10.1016/j.jss.2007.07.038.