El método de medición de tamaño funcional COSMIC

Versión 4.0.1

Introducción al método COSMIC para medir el tamaño del software

Versión 1.1

Enero 2016

Reconocimientos

Versión 1.0 y 1.1 Revisores		
Alain Abran École de Technologie Supérieure, Université du Québec Canadá	Diana Baklizky Métricas Brasil	Lindsay Davies Editor independiente Reino Unido
Peter Fagg Pentad Reino Unido	Cigdem Gencel Free University of Bolzano-Bozen Italia	Arlan Lesterhuis [1)] Países Bajos
Bernard Londeix Telmaco Reino Unido	Hassan Soubra [2)] École Supérieure des Techniques Aéronautiques et de Construction Automobile Francia	Charles Symons [1)] Reino Unido
Mónica Villavicencio ESPOL Ecuador	Frank Vogelezang [2)] Ordina Países Bajos	Chris Woodward CW Associates Reino Unido

Equipo de traducción de la Versión		
Francisco Valdés Souto SPINGERE Mexican Software Metrics Association (AMMS) National Autonomous University of Mexico - Science Faculty México		

1) Editores de este documento.

2) Revisores de la v1.1

Control de versiones

La siguiente tabla resume la evolución de este documento en su versión inglesa.

FECHA	REVISOR(ES)	Modificaciones / Adiciones
2000	Miembros del equipo COSMIC	"Introducción y Resumen" Presentación de diapositivas y notas adicionales.
Septiembre 2007	14 revisores de 7 países	Primera versión del documento "Resumen del método" para la v3.0 del método COSMIC.
Mayo 2014	Comité de prácticas de medición COSMIC	Versión 1.0 de la "Introducción al método COSMIC para medir el software", para la versión 4.0 de COSMIC.
Enero 2016	Comité de prácticas de medición COSMIC	Modificado para hacer referencia al nuevo sitio web sobre "medición COSMIC" y a la versión 4.0.1 del método COSMIC. Se realizaron también otras correcciones y cambios editoriales. Véase el apéndice A.

Prefacio

El método COSMIC es un método estandarizado internacionalmente (ISO 19761, véase [1]) para medir el tamaño de los requerimientos funcionales de la mayoría de los dominios de software, incluyendo software para aplicaciones de negocios (o "sistema de información administrativo"), software de tiempo real, software de infraestructura y algunos tipos de software científico/ingeniería.

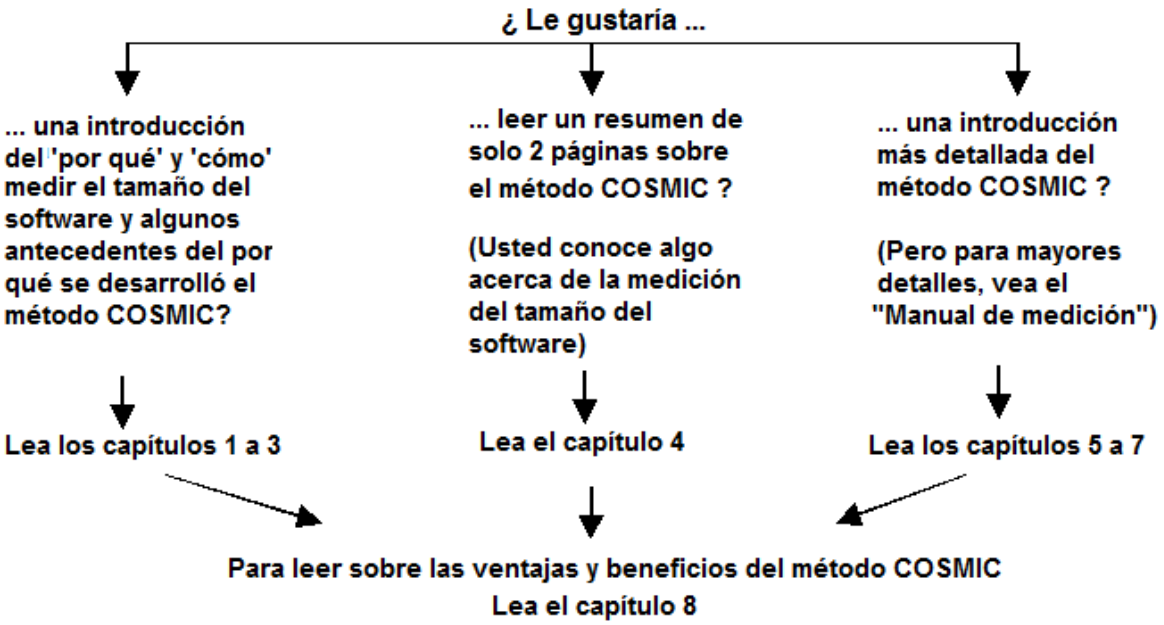
"COSMIC" son las siglas en inglés de "Common Software Measurement International Consortium". Creado en 1998 por un grupo de expertos de medición del software de Australia, Europa y Norteamérica, con la finalidad de desarrollar un nuevo método para medir el tamaño del software con base en principios de ingeniería del software bien establecidos y criterios de metrología. Sus publicaciones están completamente abiertas y disponibles para su descarga gratuita.

Hoy en día, el método es ampliamente utilizado en todo el mundo, en todos los dominios para los que se diseñó, para fines tales como medir el tamaño en los contratos de software, y se aplica de manera exitosa para medir el desempeño del proyecto, la evaluación comparativa (*benchmarking*) y para la estimación.

Finalidad de este documento de "Introducción"

Este documento está dirigido a las personas que necesitan de una introducción a la medición de tamaño del software y sus usos, y que requieren de una visión general del método COSMIC, pero no a detalle.[1]

Utilice el siguiente diagrama para decidir que capítulos leer.



Documentación del documento COSMIC

Toda la documentación COSMIC excepto el estándar ISO 19761 pueden descargarse de la base de conocimiento del sitio web COSMIC www.cosmic-sizing.org.

Los principales documentos que definen el método son:

[1] La introduccion reemplaza el documento "Resumen del método" y aplica a la versión 4.0.1 del método.

- La norma ISO 19761 ('Ingeniería de software – COSMIC – Un método de medición de tamaño funcional'), que contiene las definiciones y reglas básicas del método. (Al momento de escribir este documento, la versión 2012 de esta norma aún no se actualiza la v4.0.1 del método).
- La versión 4.0.1 del método COSMIC: "Manual de Medición" que proporciona todos los principios y reglas, así como el glosario de términos, también explica con mayor detalle y con muchos más ejemplos el método, con el fin de ayudar a los que miden a entender y aplicar el método. Este es el principal "documento de trabajo" que el medidor (la persona que miden o el responsable de la medición) necesitará en la práctica.

La figura a continuación muestra la estructura de la documentación de COSMIC. Al momento de la publicación de esta Introducción, la mayoría de estos documentos se encuentran disponibles; se indica con un asterisco el documento que aún se encuentra bajo desarrollo. Todos han sido actualizados para estar en línea con la versión 4.0.1 del método.

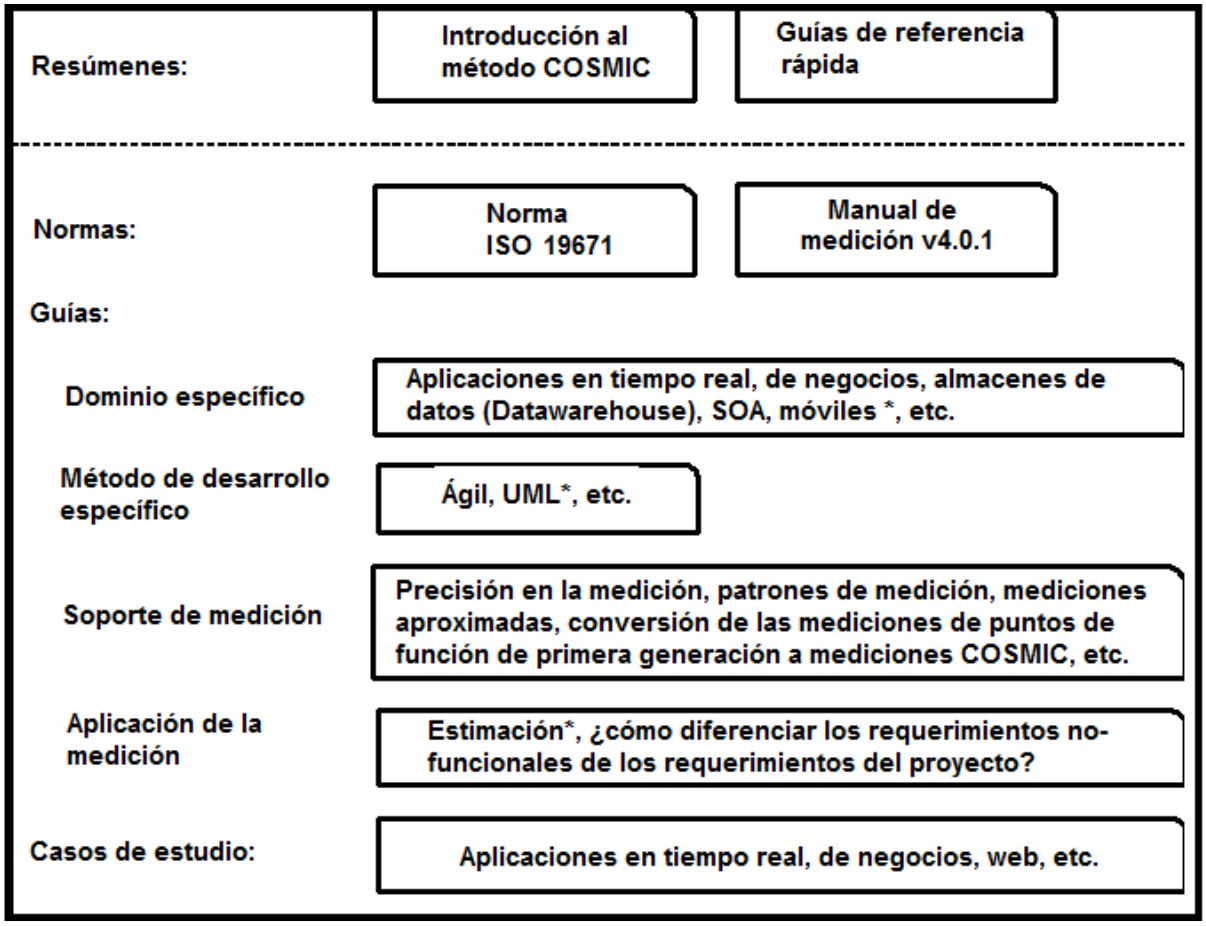


Figura – Estructura de la documentación COSMIC

Las traducciones del "Manual de Medición" también se encuentran disponibles en 11 idiomas además del inglés. Todos estos pueden encontrarse en el sitio web de COSMIC www.cosmic-sizing.org.

Este mismo sitio web contiene más información más general sobre la medición del tamaño funcional y sus usos, de la organización COSMIC y de sus actividades, sobre los proveedores de servicios relacionados con COSMIC, exámenes de certificación COSMIC, boletines COSMIC, cómo contribuir y obtener datos de evaluaciones comparativas (*benchmarks*), etc., así como de herramientas que apoyan la medición y muchos artículos de investigación relacionados con COSMIC, todos para su descarga gratuita.

El comité de prácticas de medición COSMIC

Enero 2016

Contenido

-

1

¿POR QUÉ MEDIR EL SOFTWARE?

1.1 ¿Por qué alguien querría "medir" el software?

La razón más frecuente por la que alguien decide medir el tamaño de algún software es la necesidad de estimar el esfuerzo para su desarrollo. Un primer pensamiento puede ser "¿Qué tan grande es el software?" El tamaño del software suele ser el principal parámetro para conocer la cantidad de trabajo de desarrollo que se requiere.

Analogía: Si usted le solicita a un proveedor estimar el esfuerzo por un trabajo como es el colocar azulejos en las paredes de un baño, el proveedor primero querrá conocer la superficie de las paredes del baño (es decir, el tamaño) en donde se va a colocar el azulejo. Entonces, conociendo la velocidad promedio con la que coloca los azulejos del tamaño requerido, por ejemplo, azulejos por hora (lo que se denomina "productividad") el proveedor puede darle a usted una primera estimación del esfuerzo. El punto de partida para tal estimación es siempre

Esfuerzo estimado = Tamaño estimado entre productividad.

Esta estimación inicial podría refinarse debido a los rincones o ventanas inusuales en el baño, pero el principal "factor de costo" para el esfuerzo es el área (= tamaño estimado) a colocar el azulejo.

El proceso de estimación es similar al estimar el esfuerzo para desarrollar o actualizar algún software. Es necesario medir o estimar el tamaño del software que se va a desarrollar o actualizar. Se pueden obtener datos de productividad de proyectos de desarrollo de software que utilizan tecnología similar a la que se utilizará para el nuevo software.

- de las mediciones de la productividad de proyectos terminados dentro de su propia organización,
- de fuentes de datos en los *benchmarks* como lo es la base de datos de la industria ISBSG disponible públicamente en www.isbsg.org.

(Por cierto, tenga cuidado de distinguir el "tamaño del software", que es el tema de esta Introducción, del "tamaño del proyecto". Un proyecto puede incluir otras actividades además del desarrollo de software; el tamaño del proyecto se mide en unidades de esfuerzo como son las "horas de trabajo", o nivel del personal o la duración).

1.2 Medir el software tiene muchos otros usos

Medir el tamaño del software puede ser muy útil para muchos otros propósitos, además de la estimación del proyecto. Por ejemplo:

- **Comparación**. Es posible que una organización desee comparar la productividad mediante un enfoque Ágil para la administración de proyectos con el enfoque tradicional de "cascada". Para ello debe utilizar el mismo método de medición de software desarrollado para todos los tipos de proyectos.
- **Controlar cambios al alcance, presupuesto y progreso**. El seguimiento del tamaño de una nueva parte del software a medida que evolucionan sus necesidades ayuda a los administradores de proyectos a controlar los "cambios al alcance" y, por lo tanto, el presupuesto del proyecto, así como controlar el progreso con respecto al presupuesto.
- **Controlar densidad de defectos.** Cuando se termina un proyecto, es posible que se desee detectar el número de defectos encontrados en el primer mes de funcionamiento e informar, por

ejemplo, la “densidad de defectos” en términos de defectos por unidad de tamaño. Véase el capítulo 8 para más usos de la medición del tamaño del software.

.

1.3 ¿Quién se beneficia, por lo general, de estas mediciones?

Los principales proveedores comerciales de software miden de manera rutinaria el tamaño de los programas y los utilizan para estimar el nuevo proyecto y para medir la productividad del proyecto. Sus medidas son vitales para administrar el riesgo y mantener la rentabilidad. Los clientes de software pueden beneficiarse aún más de usar estas mediciones para controlar los cambios al alcance, el precio/desempeño de los proveedores, calidad de los entregables, etc.

2

¿CÓMO MEDIR EL TAMAÑO DEL SOFTWARE?

2.1 Como cualquier otra unidad de medida, necesita normas

El tamaño del software puede medirse de muchas formas y en diferentes fases del ciclo de vida del proyecto de software.

Si usted desea utilizar el tamaño del software para varios propósitos a través de múltiples actividades, es evidente que debe adoptar una forma estándar de medir el software. El método COSMIC es un ejemplo de un Método de Medición de Tamaño Funcional (FSM, por sus siglas en inglés) estandarizado internacionalmente que ha sido aceptado como la Norma Internacional ISO 19761.

2.2 ¿Cuáles son los métodos más importantes para medir el software?

Existen tres métodos principales para medir el tamaño del software.

- Puede contar las líneas de código fuente (SLOC; por sus siglas en inglés) escritas para implementar los requerimientos del software.
- Puede medir el tamaño de los requerimientos del software.
- Puede utilizar un método relacionado con la etapa del desarrollo del software.

2.3 Contar las líneas de código fuente

Contar SLOC fue uno de los primeros métodos para medir el tamaño del software. La ventaja de medir con SLOC es que se puede contar de manera automática con programas que analizan el código fuente, aunque el tamaño en SLOC tiene desventajas significativas.

- No existen normas universales para contar SLOC, los resultados pueden variar de un programa automático que cuenta código fuente a otro.
- Cuando se programa un conjunto dado de requerimientos de software, los números de SLOC dependerán del lenguaje de programación utilizado y hasta de la habilidad del programador. Las comparaciones de productividad entre proyectos, especialmente cuando se utilizan lenguajes de programación diferentes son, por lo tanto, intrínsecamente difíciles.
- Solo se conoce el tamaño en SLOC cuando se termina de codificar el programa de software. Por lo tanto, es difícil utilizar el tamaño en SLOC para estimar el esfuerzo en fases tempranas de la vida de un proyecto. Para estimar un tamaño en SLOC, el proyecto debe haber progresado hasta un punto en el que ya se tiene conocimiento del diseño y la estructura del programa y, entonces se necesitará de experiencia para hacer conjeturas o analogías.
- Con algunos lenguajes y herramientas de programación es difícil identificar líneas de código fuente, sobre todo en aquellos lenguajes que se basan en seleccionar y establecer parámetros y opciones.

Sin embargo, el tamaño en SLOC se sigue utilizando cuando el tamaño físico del software es relevante y en algunos dominios de software que han acumulado años de experiencia en el uso de esta medida. Además, varios métodos bien conocidos para estimar proyectos de software, por ejemplo, COCOMO II [2], están calibrados mediante tamaños en SLOC.

2.4 Medir los requerimientos del software

Los métodos para medir los requerimientos funcionales del software, conocidos como métodos de "Medición de Tamaño Funcional" (o "FSM") uno de los cuales es COSMIC, tienen la clara ventaja de que se pueden utilizar poder estimar proyectos tan pronto se conocen los "Requerimientos Funcionales del Usuario" (FUR, por sus siglas en inglés). La mayoría de los métodos FSM también tiene variantes que se pueden utilizar para aproximar el tamaño incluso antes de que se conozcan en detalle los FUR.

La otra gran ventaja de los métodos FSM es que los tamaños obtenidos son independientes de la tecnología utilizada para implementar el software. Además, para algunos de los métodos FSM, sus unidades de medida están estandarizados internacionalmente. Las unidades de medida de los métodos FSM son el equivalente más cercano que la industria de software tiene como unidad estándar (como lo es el metro para medir longitud).

2.5 Otras formas de medir el software

Existen muchos otros métodos para medir el tamaño del software, pero casi todos ellos están relacionados con métodos de desarrollo específicos o para medir el tamaño en una etapa particular del desarrollo.

Algunos ejemplos incluyen los "Puntos por caso de uso" de UML, "Puntos por historia de usuario" para los métodos ágiles, "Puntos objeto" para los métodos orientados a objetos, etc. Ninguno de estos métodos está bien definido y soportado por grupos de usuarios internacionales, o son portables a través de diferentes métodos de desarrollo. Ninguno ha sido estandarizado internacionalmente. Algunos, como los "puntos por historia de usuario', son realmente muy subjetivos.

2.6 Una perspectiva más detallada de los requerimientos de software – FUR y NFR

Una perspectiva más detallada de los requerimientos de software muestra que hay dos tipos de requerimientos: "Requerimientos Funcionales del Usuario" (o "FUR", por sus siglas en inglés) y "Requerimientos No Funcionales" (o "NFR", por sus siglas en inglés). En términos más sencillos:

- FUR establecen lo que el software debe hacer para los usuarios, en términos de tareas y servicios (ISO 14143-1);
- NFR son, por lo general, restricciones que aplican a todo el sistema hardware/software.

En el siguiente ejemplo se muestran tanto los FUR como los NFR:

EJEMPLO DE UN SISTEMA DE NEGOCIOS: Suponga un sistema de personal de una compañía.

- *Los FUR especificarán que se debe permitir la entrada y mantenimiento de todos los datos acerca de los empleados de la compañía, incluyendo su nombre y dirección, fecha de nacimiento e inicio de empleo, grado, cargo, departamento, títulos académicos, dependientes, progreso en su carrera y registro de evaluación, etc. El software también debe considerar consultas sobre los datos almacenados.*
- *Los NFR para este mismo sistema de personal podrían especificar: controles de seguridad en el acceso, disponibilidad del sistema, la tecnología a utilizar para el software, tiempo de respuesta objetivo, y similares.*

EJEMPLO DE UN SISTEMA EN TIEMPO REAL: Suponga un software integrado que controla las funciones de una copiadora simple.

- *Los FUR especificarán que debe aceptar todos los comandos de usuario, por ejemplo, inicializar el sistema después de encender la copiadora, responder al usuario que introduce el número de copias requeridas, la selección del color de la copia, en negro o de color, ampliación, etc. y entonces controlar estos comandos para producir las copias después de que el usuario oprime el botón "Iniciar". El software debe también responder a los sensores que señalan cuando hay un atasco de papel, se ha terminado el papel o la tinta, etc.*
- *Los NFR para la copiadora especificarían: restricciones del tiempo del sistema, un objetivo de cero defectos para el software, criterios de disponibilidad del sistema, etc.*

Observe que, a medida que el proyecto avanza, muchos requerimientos del sistema que inicialmente aparecen como no funcionales se convierten en requerimientos funcionales del software.

EJEMPLO: Los requerimientos del sistema para auditar o para una facilidad de uso pueden surgir en las fases tempranas de un proyecto como no funcionales, pero a medida que el proyecto avanza, se traducirán en requerimientos funcionales de software que pueden ser medidos por el método COSMIC de la misma manera que cualquier otro FUR:

Una "Guía de requerimientos no funcionales y de proyecto" aconseja cómo considerar este tipo de requerimientos para medir el desempeño del proyecto de software, el *benchmark* y la estimación [14].

2.7 Habilidades necesarias para los medidores COSMIC

Para medir con precisión el tamaño del software con el método COSMIC, se necesitan las habilidades de cualquier ingeniero de requerimientos o analista de sistemas. Además, es muy deseable que se tenga conocimientos básicos de métodos estadísticos para utilizar los resultados de las mediciones de tamaño COSMIC al hacer comparaciones de desempeño del proyecto, en el desarrollo del *benchmarking* y en la estimación.

3

UNA BREVE HISTORIA DE LA MEDICIÓN DE TAMAÑO FUNCIONAL

En este capítulo se describe por qué y cómo se desarrolló el método COSMIC.

3.1 ¿Cómo inició todo?

A mediados de los 70s, Allan Albrecht de IBM se encargó de medir la productividad de los proyectos de software en una parte de IBM que empezaba a hacer uso de múltiples lenguajes de programación. Dadas las desventajas del uso de SLOC como una medida del tamaño del software entregado, tuvo la gran idea de desarrollar una medida de los requerimientos del software independiente de la tecnología utilizada.

El método de Albrecht fue publicado por primera vez en 1979 (véase referencia [3]) y se le conoce como “Análisis de Puntos de Función” (“FPA”, por sus siglas en inglés). Los encargados de la administración del desarrollo del método fue el Grupo Internacional de Usuarios de Puntos de Función (IFPUG, por sus siglas en ingles) y al método se le conoce como “IFPUG-FPA”.

Aunque el método IFPUG es todavía el método FSM más utilizado en el dominio del software de aplicaciones de negocios, este método tiene varias debilidades de las cuales se destacan las siguientes:

- Se ha vuelto cada vez más difícil asignar los tipos de funciones de Albrecht a las nuevas maneras de modelar los requerimientos de software. Esto se aplica, de manera especial, a las áreas en las que el software se construye como servicios y en los dominios del software de infraestructura y en tiempo real.
- Los tipos de funciones que considera pueden tener solo un rango muy limitado de tamaños, lo que significa que el método es insensible a los extremos de tamaño que existen en el software real. Una escala de medición debería ser normalmente lineal y abierta.

3.2 La Organización Internacional de Normalización (ISO)

Alrededor de 1990, existía la demanda de una norma ISO para el método FSM. Pero no hubo un acuerdo para que cualquiera de los métodos entonces existentes (el IFPUG y otros) fueran candidatos adecuados. El ISO, por lo tanto, estableció un grupo de trabajo[2] para estudiar y definir los principios de FSM. En 1998 se publicó una primera versión de la norma resultante, ISO 14143/1 (véase referencia [4]).

Los nuevos principios ayudaron a mejorar la comprensión de los FSM, pero no resolvieron el problema de insatisfacción con los métodos existentes. El mercado necesitaba un nuevo método FSM.

3.3 COSMIC se pone en marcha

Los procedimientos ISO fueron diseñados para obtener un acuerdo sobre las normas del conocimiento existente, pero no para el desarrollo de nuevas ideas. Por lo tanto, a fines de 1998, un grupo informal de expertos en medición del software de Australia, Europa y Norteamérica decidió desarrollar un método de segunda generación con base en los principios de la norma ISO 14143/1. El

[2] ISO JTC1/SC7/WG12

grupo se llamó “COSMIC”, por las iniciales de Common Software Measurement International Consortium.

Objetivos de COSMIC
Los objetivos del grupo COSMIC fueron desarrollar y ganar aceptación en el mercado de un método para medir los requerimientos funcionales del usuario del software con base en principios de ingeniería de software fundamentales y conforme a la teoría de la medición, aplicable para medir software de negocios, tiempo real e infraestructura.

COSMIC sigue siendo un grupo internacional completamente voluntario de expertos en medición de software tanto de la industria como del mundo académico.

COSMIC continúa perfeccionando la definición y explicación del método a la luz de la experiencia práctica, aunque debe enfatizarse que los ***principios básicos de la medición del tamaño no han cambiado desde que el método fue publicado por primera vez en 1999***.

3.4 La última palabra de ISO: “Permita que el mercado decida”

A principios de los años 2000, a nivel ISO, aun no existía un método FSM con norma internacional y no se tenía un acuerdo para que cualquiera de los métodos existentes pudiera aceptarse como tal. Finalmente, ISO acordó una política de “permita que el mercado decida”. De modo que ahora se puede elegir entre cinco métodos FSM con norma ISO (IFPUG, COSMIC y otros tres).

El método COSMIC con norma ISO (ISO 19761) fue publicado por primera vez a inicios de 2003. La última versión de esta norma, la del 2011, puede obtenerse desde el sitio web www.iso.org.

4

UN MUY BREVE RESUMEN DEL MÉTODO COSMIC

La finalidad de este capítulo es ofrecer un primer resumen a muy alto nivel del método COSMIC.

El primer uso de las palabras clave del método COSMIC en cualquier capítulo está en **negritas**. Para obtener una definición formal de estas palabras clave vea el glosario del Manual de Medición [5].

4.1 Aplicabilidad del método

El método COSMIC fue diseñado para medir los Requerimientos Funcionales del Usuario (FUR) de aplicaciones de negocio (o “sistemas de información de administración”), software de tiempo real e infraestructura y algunos tipos de software científico/ingeniería, en cualquier capa de una arquitectura de software y en cualquier nivel de descomposición del software.

4.2 Las tres fases del proceso de medición del tamaño funcional COSMIC

En la figura 4.1 se muestra el proceso de medición COSMIC. Las tres fases se explican en las siguientes secciones.

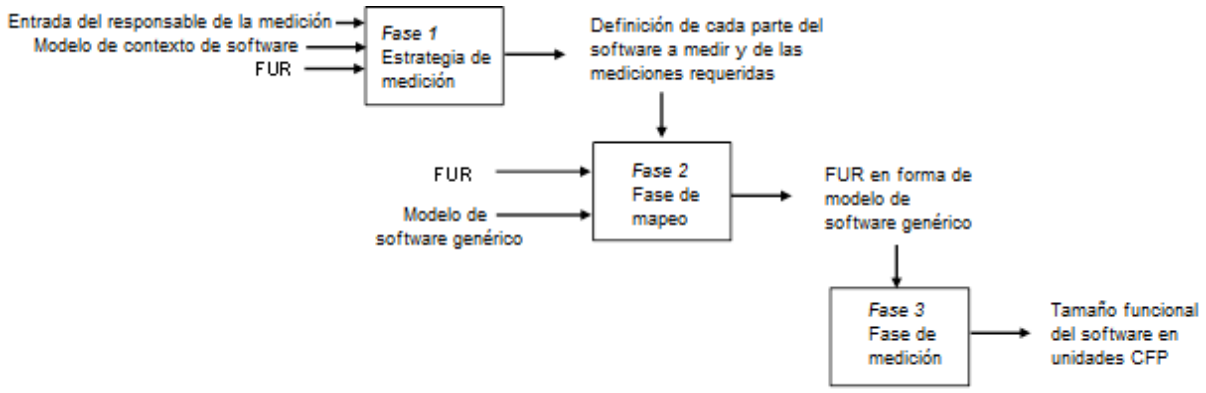


Figura 4.1 – El proceso de medición COSMIC

4.3 Fase 1: Estrategia de medición

Primero se debe definir lo qué se medirá. El tamaño de una parte del software depende del punto de vista de quién, o qué se definirá como sus **usuarios funcionales**, es decir, los humanos, dispositivos de hardware u otro software que interactúa con el software a medir. Para medir el tamaño de la parte de software se debe, por lo tanto, acordar primero el **propósito** de la medición, que lleva a definir lo que es su **alcance** (el alcance de los FUR del software a medir) y sus usuarios funcionales, y luego otros parámetros[3].

Es imprescindible documentar los parámetros de la estrategia de medición para que las medidas resultantes sean interpretadas de manera correcta por todos los futuros usuarios.

[3] Los principios subyacentes a los parámetros necesarios para la estrategia de medición están definidos en el "Modelo de Contexto de Software" de COSMIC. Este modelo no se describe en esta Introducción. Consulte el Manual de Medición.

4.4 Fase 2: Mapeo

La tarea en la fase de Mapeo es crear el modelo COSMIC de los FUR, a partir de cualquier artefacto de software disponible, por ejemplo, un esbozo o los requerimientos detallados, modelos de diseño, el software físico instalado, etc. Para crear el modelo, aplicamos los principios del **Modelo de Software Genérico** COSMIC a los FUR a medir.

Este modelo de los FUR del software se basa en cuatro principios principales:

1. La funcionalidad del software consta de **procesos funcionales**. La tarea de cada proceso funcional es responder a un **evento** que ha sucedido en el mundo de los **usuarios funcionales** del software.
2. Los procesos funcionales constan de subprocesos. Éstos hacen solo dos cosas: se mueven y manipulan los datos. Los subprocesos de **movimiento de datos** que mueven datos de usuarios funcionales en un proceso funcional y que trasladan datos a ellos se llaman **Entradas** y **Salidas**, respectivamente. Los subprocesos de movimiento de datos que mueven datos hacia y desde el **almacenamiento persistente** se les llama **Escritura** y **Lectura**, respectivamente, En la figura 4.2 se muestra los cuatro tipos de movimientos de datos.

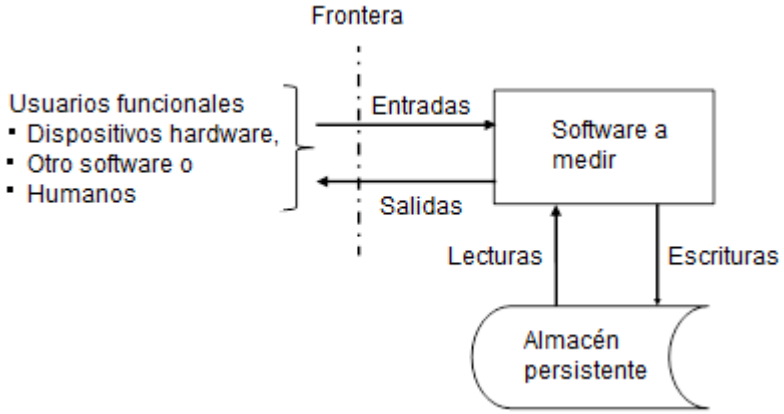


Figura 4.2 – Los cuatro tipos de movimientos de datos

3. Cada uno de los movimientos **de datos** (Entrada, Salida, Lectura o Escritura) mueve un solo **grupo de datos** cuyos a**tributos**[4] describen una sola "cosa" (un o**bjeto de interés**).
4. Se supone que los subprocesos de manipulación de datos son contados por el movimiento de datos con el que están asociados. La manipulación de datos no se mide por separado.

Un proceso funcional termina de ejecutarse cuando ha hecho todo lo que se requiere hacer para responder a los datos que recibió sobre el evento.

4.5 Fase 3: Medición

La unidad de medida del método COSMIC es el 'Punto de función Cosmic' (CFP, por sus siglas en inglés). Cada uno de los movimientos de datos es medido como 1 CFP.

En la fase de medición, medimos el tamaño de una nueva parte del software identificando todos los movimientos de datos (Entradas, Salidas, Lecturas y Escrituras) de cada uno de los procesos funcionales y se suman estos en todos sus procesos funcionales.

Un proceso funcional debe tener al menos dos movimientos de datos (una Entrada y una Salida o una Escritura) para proporcionar un servicio mínimo pero completo. Por lo tanto, el tamaño mínimo de un proceso funcional es 2 CFP. No hay un límite superior para el tamaño de un proceso funcional.

[4] Conocido como 'Tipos de Elementos de Datos' o 'DETs' en algunos otros métodos FSM.

Para medir una mejora del software existente, se identifican todos los movimientos de datos que se van agregar, cambiar y borrar, y se suman estos en todos sus procesos funcionales. El tamaño mínimo de cualquier modificación a un proceso funcional es 1 CFP.

5

EL MÉTODO COSMIC – LA FASE DE ESTRATEGIA DE MEDICIÓN

Medir el tamaño funcional COSMIC debe seguir un proceso de tres fases. En la primera fase de la Estrategia de Medición, el medidor debe acordar con quien necesita la medida (el "responsable") el propósito de la medición además de otros parámetros que dependen del propósito.

En este capítulo y los subsiguientes, el primer uso de una palabra clave COSMIC está en **negritas**. Las definiciones formales de las palabras clave se encuentran en el glosario del Manual de Medición (MM) que ofrece todos los principios y reglas del método COSMIC, con muchos ejemplos [5]. Esta introducción sólo contiene definiciones informales.

5.1 ¿Por qué necesitamos una 'estrategia'?

Es necesario acordar y documentar el propósito de la medición y otros parámetros con el responsable de la medición para que en el futuro todos comprendan el tamaño medido y cómo se puede usar.

En la práctica, se encontrará que sólo unos pocos "patrones" recurrentes de parámetros serán necesarios para los diferentes tipos de software a medir en su organización. Como ayuda, en la 'Guía COSMIC para Patrones de Estrategia de Medición' [6] define algunos de los patrones más comunes y sus usos.

5.2 Los cinco parámetros de estrategia a determinar

- El **propósito** de la medición. El propósito ayuda a determinar los siguientes parámetros.
- El **alcance** de la parte(s) del software a medir. Un proyecto podría consistir de la entrega de varias partes del software, o la funcionalidad a medir podría estar restringida de alguna manera. ¿Qué se incluye y qué se excluye en la funcionalidad?
- El **nivel de descomposición** de esta parte(s) del software a medir. Diferentes niveles serían, por ejemplo, una "aplicación completa" ("nivel 0"), o uno de los componentes primarios de un sistema distribuido ("nivel 1"), o un componente reutilizable en una arquitectura SOA ("nivel 2").
- Los **usuarios funcionales** de cada parte del software a medir. Estos son humanos o "cosas" (dispositivos hardware u otras partes del software) que son remitentes o destinatarios previstos de datos hacia/desde el software que se mide. Es la funcionalidad que "perciben", que se medirá;
- La **capa**(s) de la arquitectura de software en la que el software reside. Una parte del software a medir debe ser confinado a una capa.

Mediante la documentación de estos parámetros para cada tamaño medido, se ayudará a garantizar que en el futuro los tamaños sólo se compararán y se utilizarán sobre una base de "igual a igual".

5.3 "Capas" del software

La mayoría de los parámetros de la estrategia de medición son fáciles de entender. Pero el término "capa" se utiliza de diversas maneras en la industria del software. {A veces se utiliza "nivel-n" en vez de "capa-n"}. En la figura 5.1 se muestra un sistema de cómputo típico de "arquitectura por capas" que soporta un software de aplicación de negocios.

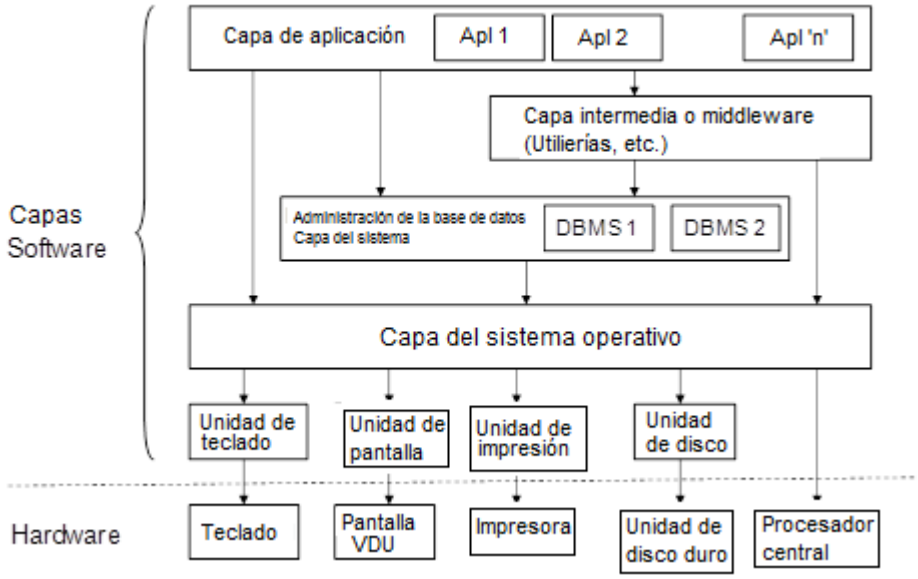


Figura 5.1 – Arquitectura de software en capas típica para un sistema de cómputo de negocios/sistema de cómputo MIS

En la figura 5.2 se muestra que la capa de aplicación de la figura 5.1 puede subdividirse en otras capas, dependiendo de la "vista" de la arquitectura de software (y en consecuencia de los usuarios funcionales del software a medir, como se ve a continuación).

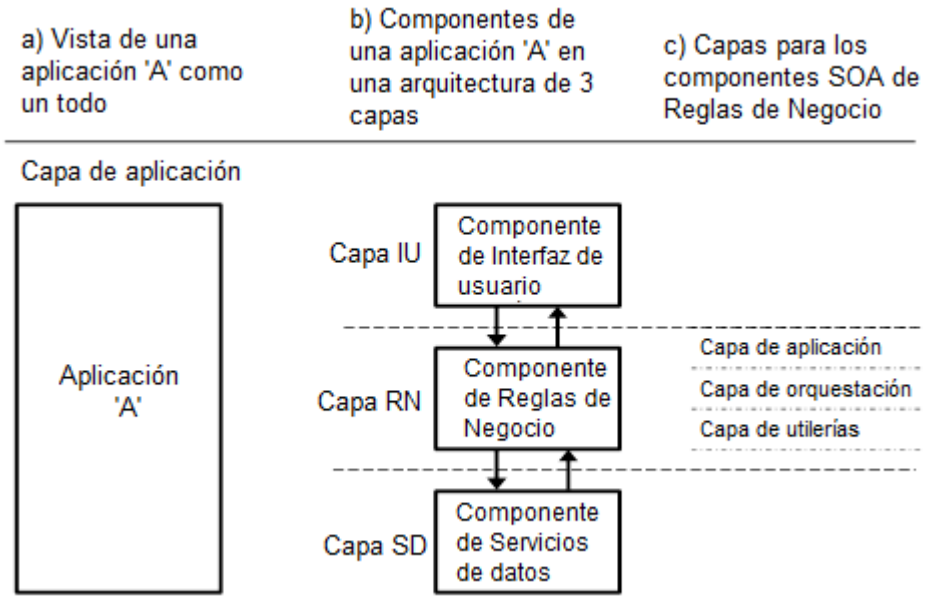


Figura 5.2 – Tres vistas de una parte del software de aplicación

5.4 Ejemplos de cómo es que el "propósito" de una medición afecta a los otros parámetros de estrategia de medición

EJEMPLO DE UN SOFTWARE DE NEGOCIOS: Suponga que el software a desarrollar y medir es un sistema de aplicación de negocios de 3-capas distribuida. El contexto es un contrato con un proveedor que estipula que, para propósitos de pago, los tamaños del software deben medirse a nivel de todas las aplicaciones, ignorando cualquier estructura de componente.

Caso 1.

Propósito: Medir el tamaño de una aplicación entregada para el pago del contrato

Alcance: Los FUR de una aplicación

Usuarios funcionales: Usuarios humanos y cualesquiera otras interfaces de aplicaciones

Nivel de descomposición: Ninguno ("nivel 0")

Capa: Aplicación, es decir, la vista a) de la figura 5.2

Caso 2.

Propósito: Medir el tamaño de cada componente principal de la aplicación distribuida para que el proveedor pueda estimar el esfuerzo del proyecto, ya que cada componente se desarrollará utilizando una tecnología diferente.

Alcance: Cada componente se mide por separado (es decir, hay tres alcances de la medición).

Usuarios funcionales: Refiérase a las tres capas como en la vista b) de la figura 5.2

El componente de Interfaz de Usuario tiene los usuarios humanos y el componente Reglas de Negocio como sus usuarios funcionales

El componente Reglas de Negocio tiene la interfaz de usuario y los componentes de servicios de datos como sus usuarios funcionales

El componente Servicios de Datos tiene el componente Reglas de Negocio y cualquier otra interfaz de aplicación como sus usuarios funcionales

Nivel de descomposición: Primer nivel de descomposición de una aplicación ("nivel 1")

Capas: Véase las tres capas de vista b) de la figura 5.2

EJEMPLO DE SOFTWARE DE TIEMPO REAL: La funcionalidad del software integrado en un dispositivo de hardware utilizado por seres humanos, por ejemplo, una combinación de impresora/copiadora en computadora puede medirse desde el punto de vista de dos tipos de usuarios funcionales. (En ambos casos supongamos que no estamos interesados en ninguna estructura de componentes del software ni en ningún firmware que el software integrado pueda utilizar).

Caso 1.

Propósito: Medir el tamaño de la funcionalidad disponible para el usuario humano (la "oferta del consumidor" para fines comerciales), para compararlo con la oferta de productos competitivos.

Alcance: La funcionalidad disponible para los usuarios de los operadores humanos (es decir, excluyendo la funcionalidad necesaria para los operadores sobre los que no tienen control o no pueden "percibir", como algunas funciones necesarias para comunicarse con la computadora)

Usuarios funcionales: Usuarios operadores humanos

Caso 2.

Propósito: Medir la funcionalidad que el desarrollador del software integrado debe proporcionar para que el dispositivo funcione

Alcance: Toda la funcionalidad del software integrado

Usuarios funcionales: Todos los dispositivos de hardware con los que el software debe interactuar (por ejemplo, teclado, botones de control, pantallas, mecanismo de unidad de impresión, mecanismo de transporte de papel, etc., cualquier computadora con la que la impresora debe comunicarse y el software del controlador de la impresora).

5.5 ¿Qué más debería pensar antes de iniciar la medición?

Es muy importante determinar lo artefactos del software que están disponibles a usar para determinar los FUR que se van a medir. En la práctica, los artefactos disponibles pueden no suministrar exactamente la información necesaria para cualquier medida FSM, por lo que usualmente el medidor tiene que hacer algunas suposiciones para determinar los FUR. Lo mejor es consultar a un experto en los requerimientos del software que se va a medir para que ayude a entender el sistema y que la medición sea lo más precisa posible.

Algunos ejemplos de problemas que se enfrentan con frecuencia:

- Si se necesita medir el tamaño al principio de la vida de un proyecto, es posible que aún no se hayan documentado los requerimientos a detalle que son necesarios para obtener una medida precisa de COSMIC. Para estas situaciones, tenemos una Guía [7] que describe las variantes del método COSMIC estándar que se puede utilizar para medir el tamaño aproximado;
- A veces, los requerimientos de software se definen a un alto nivel y luego en niveles cada vez más detallados. Llamamos a estos: **niveles de granularidad**. Para garantizar la comparabilidad, los tamaños deben medirse en el nivel estándar de granularidad de los “procesos funcionales” (véase más adelante). Si es necesario, se puede utilizar una variante para aproximar el tamaño desde un nivel más alto de granularidad hasta el nivel estándar;
- A veces se debe medir el tamaño de un sistema instalado para el cual ya no existen los requerimientos. En estos casos, para determinar los FUR, el medidor necesitará "llevar a cabo una ingeniería inversa" a partir de los artefactos disponibles, por ejemplo, pantallas, documentación del usuario, informes, interfaces de usuario, etc.

Varias Guías COSMIC describen cómo reconocer o analizar FUR para diferentes tipos de software o métodos de desarrollo. Todos ellos disponibles en el sitio web www.cosmic-sizing.org..

Por último, se le puede solicitar al experto en medición que estime cuánto tiempo le llevará medir una parte particular de software. La velocidad media de medición utilizando el método COSMIC es similar a la de otros métodos estándar FSM. Pero la velocidad real puede variar mucho sobre este promedio. El esfuerzo necesario para medir se incrementará:

- si la calidad de los artefactos disponibles para medir es muy mala;
- si el responsable de la medición necesita de una mayor precisión de la medición y el nivel de detalle de medición debe ser documentada;
- si el responsable de medir tiene menos experiencia en el tipo de software que va a medir y en el método COSMIC.

6

EL MÉTODO COSMIC – LA FASE DE MAPEO

El propósito de la fase de Mapeo es producir un modelo de los Requerimientos Funcionales del Usuario (FUR) del software a medir a partir de los artefactos disponibles utilizando los principios del "**Modelo de Software Genérico"** de COSMIC. Primero se exponen estos principios, luego se describen los elementos del modelo con mayor detalle y por último se describe el proceso de mapeo

6.1 El Modelo de Software Genérico

PRINCIPIOS – El Modelo de Software Genérico COSMIC
a) Una parte del software que interactúa con sus usuarios funcionales a través de una **frontera**, y con un **almacenamiento persistente** dentro de esta frontera
b) Los requerimientos funcionales del usuario de una parte del software a medir pueden ser mapeados a un único proceso funcional
c) Cada uno de los procesos funcionales consta de subprocesos
d) Un subproceso puede ser, ya sea un **movimiento de datos** o una **manipulación de datos**
e) Existen cuatro tipos de movimientos de datos, **Entrada**, **Salida**, **Escritura**, **Lectura**. Una Entrada mueve un grupo de datos en un proceso funcional de un usuario funcional. Una Salida mueve un grupo de datos de un proceso funcional a un usuario funcional. Una Escritura mueve un grupo de datos de un proceso funcional a un almacenamiento persistente. Una Lectura mueve un grupo de datos del almacenamiento persistente a un proceso funcional.
f) Un movimiento de datos mueve a un solo **grupo de datos**
g) Un grupo de datos consta de un único conjunto de **atributos de datos** que describen a un solo **objeto de interés**
h) Cada proceso funcional se inicia por su movimiento de datos de **Entrada desencadenante**. El grupo de datos movido por la Entrada desencadenante lo genera un usuario funcional en respuesta a un **evento desencadenador**.
i) Un proceso funcional deberá incluir al menos un movimiento de datos de Entrada y un movimiento de datos de Escritura o de Salida, es decir, deberá incluir un mínimo de dos movimientos de datos. En un proceso funcional no hay límite superior al número de movimientos de datos
j) Como aproximación para fines de medición, los subprocesos de manipulación de datos no se miden por separado; se supone que la funcionalidad de cualquier manipulación de datos se explica por el movimiento de datos con el que está asociado

Comentario importante: TODAS las palabras clave COSMIC en los principios anteriores (excepto "almacenamiento persistente" deberían iniciar con la palabra "tipo". Por ejemplo, los "subprocesos" deberían escribirse realmente como "tipo de subproceso" y "Entrada" como "tipo de Entrada". TODOS los métodos FSM intentan identificar "tipos" y no "ocurrencias" de funciones o datos. Sin embargo, se omite "tipo" de todas estas palabras clave para facilitar la lectura, a menos que sea necesario distinguir "tipos" y "ocurrencias".

El principio a) solo resume todo lo que hace el software. Los otros principios se explican en las secciones siguientes de este capítulo.

6.2 Una relación clave: eventos/usuarios funcionales/procesos funcionales

Los principios b) y h) nos dicen que la tarea del software es responder a los eventos que ocurren en el mundo de sus usuarios funcionales. Un usuario funcional informa al software que se ha producido un evento y puede enviar datos sobre ese evento. El software debe hacer algo útil para el (los) usuario(s) funcional(es) que tienen interés en responder a ese evento. Llamamos a esto "algo útil" un "proceso funcional". Todo los FUR del software puede expresarse en términos de procesos funcionales.

En la figura 6.1 se muestra las relaciones entre los eventos en el mundo de los usuarios funcionales y los procesos funcionales del software. (La **frontera** es la interfaz entre el software a medir y su(s) usuario(s) funcional(es)).

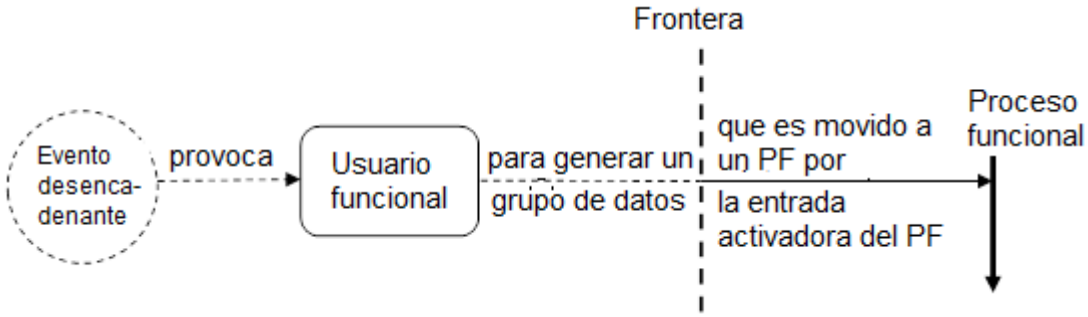


Figura 6.1 – Las relaciones entre eventos, usuarios funcionales y procesos funcionales

La interpretación general de este diagrama es que *un evento hace que un usuario funcional genere un mensaje (un grupo de datos) que se mueve por una "Entrada desencadenante" a su proceso funcional, y, por lo tanto, inicia el proceso funcional.* (Observe, sin embargo, que cuando un usuario funcional humano decide hacer una consulta sobre datos existentes, el usuario humano *genera*, de manera efectiva, el evento y luego el mensaje).

Un evento es "algo que sucede". Un evento desencadenador ha sucedido o no ha sucedido; no pueden ser subdividido para los FUR del software a medir.

EJEMPLO: Suponga un partido de fútbol. Los FUR de tres aplicaciones de software diferentes podrían tener puntos de vista muy diferentes de los acontecimientos que ocurren en el partido.

La aplicación A permite a los reporteros ingresar los resultados de los partidos de fútbol para un periódico. El único evento que reconocen los FUR es el de "partido terminado".

La aplicación B es un sistema de "reportes en directo" que permite a un reportero estar ingresando comentarios, que se transmiten por la web a los usuarios en línea de la aplicación, sobre cualquier cosa que el reportero considere importante a destacar durante el partido, por ejemplo, el inicio del partido, un gol anotado, una falta, una lesión, etc. El único evento que los FUR reconocen es "cualquier cosa que sucede sobre la cual el reportero introduce un comentario en la aplicación B".

La aplicación C permite monitorear en tiempo real el rendimiento de los jugadores. Cada jugador lleva un dispositivo de detección de posición GPS y un monitor de ritmo cardiaco que transmite datos a intervalos regulares muy cortos. El único evento que los FUR reconocen es una "señal" de reloj que controla la transmisión de datos sobre la posición actual y la frecuencia cardiaca de cada jugador en cada "señal" a la aplicación C.

(Recuerde que para estos FUR se debería escribir el "tipo de evento". Cada uno de los tres FUR reconoce sólo un tipo de evento, pero es un tipo de evento diferente para cada FUR. Pero en términos de evento-ocurrencias, la aplicación A espera una ocurrencia para cada partido, la aplicación B espera quizás varias decenas por partido y la aplicación C espera varias decenas de miles por partido).

Observe que en la figura 6.1 no se dice nada acerca del *grado* (o "cardinalidad") de las relaciones entre los varios conceptos. Por ejemplo, un solo evento podría ser detectado por varios usuarios funcionales del mismo o partes del software diferentes (por ejemplo, un terremoto detectado por varios sensores); un usuario funcional de una parte del software puede detectar muchos tipos de eventos (por ejemplo, humanos que interactúan con el software).

La única regla que se aplica a la cardinalidad de las relaciones a lo largo de la cadena de la figura 6.1 (y es una regla muy importante) es que: "*cualquiera Entrada desencadenante de una parte del software a medir puede iniciar solo un proceso funcional en ese software"*.

6.3 La estructura de FUR y de procesos funcionales

Los principios b), c) y d) que describen la estructura teórica de los FUR, es decir, su descomposición en procesos y subprocesos funcionales, se muestran en la parte izquierda de la figura 6.2.

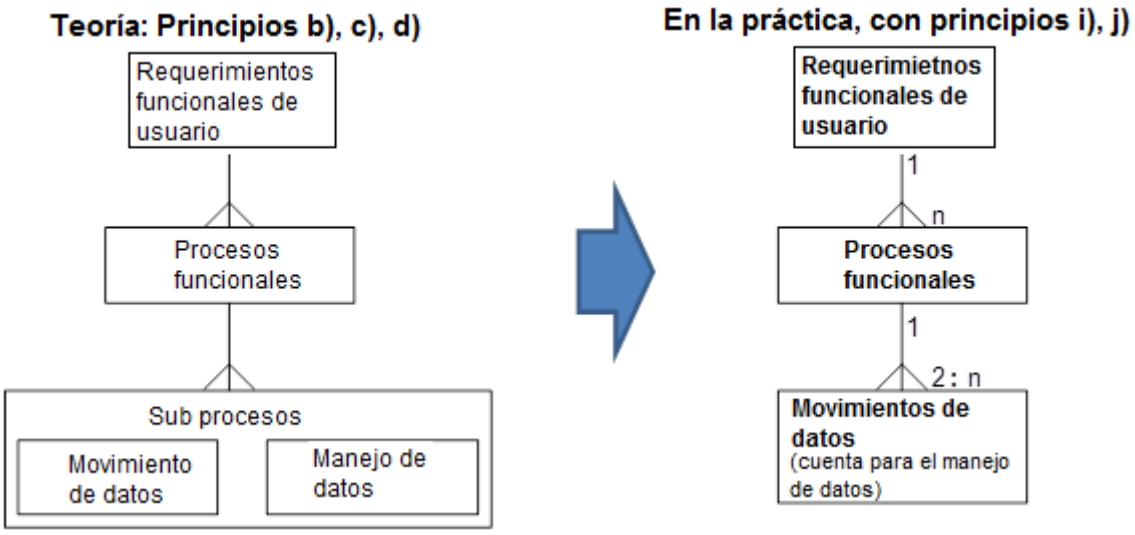


Figura 6.2 – La estructura de los requerimientos de usuario funcionales

[El símbolo de relación en forma de "pata de gallo" muestra el grado permitido de la relación entre dos conceptos adyacentes. El principio i) se expresa mostrando que un proceso funcional puede tener desde 2(mínimo) hasta "n" movimientos de datos].

La correcta identificación de procesos funcionales es el paso más importante de la fase de Mapeo. Así es que se debe entender muy bien su definición.

DEFINICIÓN – Proceso funcional
a) Un conjunto de movimientos de datos que representan una parte elemental de los Requerimientos Funcionales del Usuario para el software que se está midiendo, que es único dentro de esos FUR y que puede definirse independientemente de cualquier otro proceso funcional en los FUR.
b) Un proceso funcional puede tener solo una Entrada desencadenante. Cada proceso funcional inicia el procesamiento en la recepción de un grupo de datos movido por el movimiento de datos de Entrada desencadenante del proceso funcional.
c) El conjunto de todos los movimientos de datos de un proceso funcional es el conjunto que se necesita para cumplir con sus FUR para todas las posibles respuestas a su Entrada desencadenante.

NOTA 1: Cuando se implementa, es una *ocurrencia* de un proceso funcional que comienza a *ejecutarse* al recibir una *ocurrencia* de un grupo de datos movido por una *ocurrencia* de una Entrada desencadenante.

NOTA 2: Los FUR para un proceso funcional puede requerir una o más Entradas además de la Entrada desencadenante.

NOTA 3: Si un usuario funcional envía un grupo de datos con errores, p. ej., debido a que un usuario-sensor está funcionando mal o una orden ingresada por un ser humano tiene errores, por lo general es tarea del proceso funcional determinar si el evento realmente ocurrió y/o si los datos introducidos son realmente válidos y cómo responder a ellos.

6.4 Contar la manipulación de datos

El método COSMIC no mide la manipulación de datos de forma explícita porque no hay una manera generalmente aceptada de medir la manipulación de datos, de modo que puede combinarse con una medida de movimientos de datos para generar una medida utilizable de tamaño funcional. Por lo tanto, se invoca al principio j) para asumir que cada movimiento de datos puede explicar cualquier manipulación de datos asociada, como se muestra en la parte derecha de la figura 6.2. Esta suposición ha demostrado ser razonable para todos los propósitos prácticos, como es la medición del desempeño del proyecto y para la estimación para los que se diseñó el método y para los dominios en los que se utiliza comúnmente. Cuando el método no puede explicar la manipulación de datos, el método de medición COSMIC cuenta con extensiones locales del método para superar la limitación.

6.5 Los cuatro tipos de movimientos de datos

Principio e) como se muestra en la figura 6.3.

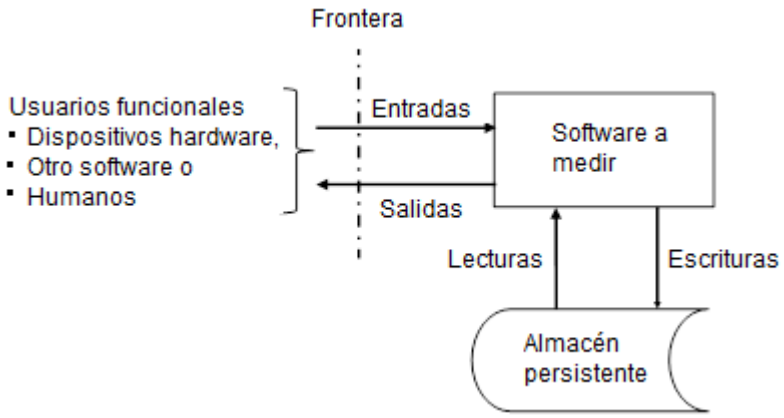


Figura 6.3 – Los cuatro tipos de movimiento de datos

Entradas y **Salidas** mueven datos hacia dentro y hacia afuera del software desde/a los usuarios funcionales, respectivamente.

Lecturas y **Escrituras** mueven datos desde el **almacenamiento persistente** al software, o viceversa, respectivamente.

6.6 Almacenamiento persistente

El **almacenamiento persistente** (mostrado en la figura 6.3) es un concepto abstracto del Modelo de Software Genérico. En este modelo, dicho almacenamiento es accesible por cualquier software en cualquier capa si se necesitan almacenar datos o recuperar datos almacenados. Después de que un proceso funcional escribe algunos datos en el almacenamiento persistente, esos "datos persistentes",

están disponibles para otros procesos funcionales que lo necesitan o para otra ocurrencia del proceso funcional que lo escribió.

Una consecuencia de este concepto es que, al medir, por ejemplo, una aplicación que debe almacenar datos o recuperar datos almacenados, no se tiene que pensar en cómo los datos se procesan de manera física por software en las capas inferiores o en el hardware. Sólo se representan los FUR que requieren que los datos sean almacenados o recuperados por la Escritura y Lectura, respectivamente.

Sólo es necesario pensar en el almacenamiento persistente en términos de unidades de disco físicos o memoria si es que debe medir software para el que se han definido dispositivos físicos de hardware (almacenamiento u otros) en la fase de Estrategia de Medición como usuarios funcionales del software. Los usuarios funcionales siempre interactúan con el software que se va a medir a través de Entradas y Salidas.

6.7 Un movimiento de datos mueve un solo grupo de datos que describe un objeto de interés

Entonces, ¿cómo distinguimos un movimiento de datos de Entrada de otro movimiento de datos de Entrada, y de manera similar para Salidas, Lecturas y Escrituras?

Un **objeto de interés** es cualquier “cosa” (física o conceptual) sobre el cual el software que se mide debe procesar o almacenar datos. Un movimiento de datos mueve un solo grupo de datos que consiste en uno o más atributos de datos (conocidos como “Tipos de Elementos de Datos” en otros métodos FSM). Todos los atributos de un grupo de datos describen el mismo objeto de interés.

No es necesario identificar los atributos de los datos con fines de medición. La única razón para mencionarlos es que a veces ayuda a distinguir los diferentes grupos de datos y sus objetos de interés examinando los atributos de datos.

En la figura 6.4 se muestra las relaciones entre estos tres conceptos, con dos ejemplos.

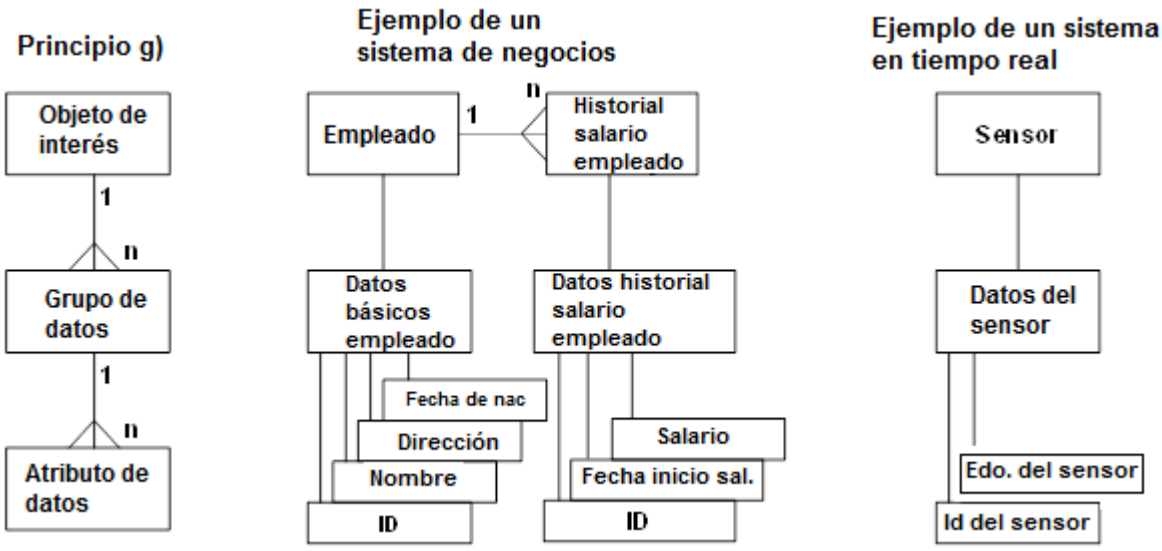


Figura 6.4 – Relaciones de un objeto de interés, grupos de datos y atributos de datos

Para ayudar a entender estos conceptos:

- Si está familiarizado con los métodos de análisis de datos, con frecuencia utilizados en el dominio de las aplicaciones de negocio, los tipos de entidad encontrados en el Análisis de Entidad-

Relación y los sujetos de relaciones en la 3ª forma normal que se encuentran en el Análisis de Datos Relacionales serán objetos de interés. Pero, por lo general, estos métodos de análisis sólo se aplican a la estructura de los grupos de datos almacenados (o "persistentes"). Para una medición COSMIC, también se necesitará aplicar estas mismas ideas de análisis para distinguir objetos de interés y, por tanto, los movimientos de datos en la Entrada y Salida de procesos funcionales. Cada Entrada y Salida mueve un grupo de datos transitorios que describen un objeto de interés.

- Existe una relación de uno a uno entre los objetos de interés y las clases de objetos resultantes del análisis UML, aunque, por supuesto, no son el mismo concepto.
- En el dominio del software de tiempo real, normalmente NO es necesario pensar en los objetos de interés. A menudo, como se muestra en el ejemplo en tiempo real de la figura 6.4, el usuario funcional - el sensor - puede verse como el envío de datos sobre sí mismo, es decir, el usuario funcional juega la parte del objeto de interés, por lo que se han identificado en la fase de la Estrategia de Medición. (No tiene sentido hacer todas estas distinciones a menos que hacerlo contribuya al proceso de medición).

6.8 El proceso de la fase de Mapeo

Suponiendo que los artefactos disponibles del software que se va a medir están a nivel del proceso funcional de granularidad, se analizan para determinar los FUR expresados como un modelo COSMIC. Los pasos de este proceso (recordando que siempre nos referimos a los *tipos*) son:

- Identificar los eventos separados en el mundo de los usuarios funcionales a los que el software debe responder, es decir, los “eventos desencadenantes”.
- Identificar qué usuario(s) funcional(es) del software debe responder a cada evento desencadenante generando un grupo de datos que es movido por una Entrada desencadenante
- Identificar un proceso funcional para cada Entrada desencadenante
- Identificar cualquier otra Entrada y todas las Salidas, Lecturas y Escrituras de cada proceso funcional necesario para cumplir con los FUR para todas las posibles respuestas a la Entrada desencadenante.

Para el último paso es posible que se necesite identificar los grupos de datos que se mueven en cada movimiento de datos y los objetos de interés que los datos describen.

6.9 Algunos ejemplos sencillos de Mapeo

Ahora podemos analizar algunos ejemplos simples para mapear los requerimientos enunciados de los requerimientos a los procesos funcionales y los movimientos de datos COSMIC utilizando el modelo de software genérico.

EJEMPLO DE UN SISTEMA DE NEGOCIOS: Un simple sistema de personal

Enunciado de los requerimientos. Se requiere un sistema para que el personal pueda mantener y actualizar datos sobre los empleados, incluyendo su salario y el historial de sus aumentos salariales en el tiempo. [El enunciado también describe los datos (atributos) que se deben registrar acerca de cada empleado y sus criterios de validación, pero en este ejemplo sencillo, la mayor parte de este detalle no necesita preocupar al Medidor]. Se requiere además de un reporte mensual con todos los empleados por su nombre y su salario actual, el número total de empleados y el costo salarial total actual.

Parámetros de la estrategia de medición

Propósito de la medición: Una medida exacta del tamaño funcional del Sistema de Personal para la estimación del esfuerzo del proyecto.

Alcance de la medición: Todo el sistema especificado en el enunciado de requerimientos.

Usuarios funcionales: El personal.

Capa: Capa de la aplicación.

Nivel de descomposición: "nivel 0", es decir, sin descomposición.

Fase de Mapeo

Algunas suposiciones:

- *Aplica la estructura de datos del ejemplo de negocios de la figura 6.4.*
- *A cada empleado se le asignará una identificación única por un miembro del personal. La llave del registro del "historial de salario de empleado" es [ID del empleado, fecha de inicio del salario].*
- *La palabra 'mantener' en el enunciado general de requerimientos, por lo general, implica que deben existir los procesos funcionales Crear, Leer, Actualizar y Eliminar (recordar el acrónimo "CRUD", por sus siglas en inglés) para cada objeto de interés. "Actualizar" permitirá un cambio de cualquier atributo, excepto el atributo llave de cualquier grupo de datos.*
- *Existen dos objetos de interés ("empleado" e "historial de salarios de empleados") sobre los que deben mantenerse los datos persistentes. Son necesarios los cuatro procesos funcionales "CRUD" para mantener los datos base de los empleados.*
- *También asumimos que se debe crear un historial de salario de empleado cuando el empleado comienza a trabajar. Posteriormente, el salario de un empleado se puede actualizar en cualquier momento, es decir, no sólo cuando se actualizan los datos base del empleado. Por lo tanto, no hay necesidad de procesos separados para "Crear" o "Eliminar" sobre el historial de sueldos de los empleados. Sin embargo, existe la necesidad de un proceso funcional para "Actualizar el salario de los empleados" y de un proceso funcional de "Lectura" separado de las consultas sobre los datos salariales de los empleados. Agregar en el proceso para producir el informe mensual significa que los requerimientos pueden ser satisfechos por 7 procesos funcionales. (A continuación, se muestra el análisis de cuatro de estos).*
- *En situaciones prácticas también puede haber FUR para un proceso funcional de "Lectura" para mostrar los datos base del empleado separado de la consulta que muestra el historial de salarios del empleado. Además, cuando un empleado se va, se puede requerir el proceso funcional "Eliminar" para archivar la base del empleado y los datos del historial de salarios, en lugar de eliminarlo. Por simplicidad, ignoramos estos posibles requerimientos.*
- *Tenga en cuenta también, como regla general, cuando se miden las aplicaciones de negocios en línea, que los menús que sólo sirven para navegación y para seleccionar procesos funcionales, así como las pantallas de entrada de datos "en blanco" deben ser ignorados. Es el movimiento de datos mediante Entradas, Salidas, Lecturas y Escrituras que deben ser identificados para fines de medición.*

1. Análisis del proceso funcional 'Crear empleado'.

El FUR es introducir datos para un nuevo empleado.

(Los ejemplos muestran los movimientos de datos y el grupo de datos que cada uno de ellos mueve).

Proceso funcional 'Crear empleado'. Evento desencadenador: se contrata a una nueva persona

Entrada desencadenante	*:*	*Datos base del empleado*
Entrada	*:*	*Salario inicial y su fecha de inicio de contratación.*
Lectura	*:*	*Datos base del empleado (para comprobar que ningún empleado ya existe con el ID ingresado)*
Escritura	*:*	*Datos base del empleado*
Escritura	*:*	*Historial de salarios de los empleados (se crea un nuevo registro cuando se ingresa el salario por primera vez)*

Salida : Mensajes de error/confirmación (Debe haber mensajes de error para varios fallos de validación y también alguna forma de confirmación exitosa de la entrada de datos. Se incluye una salida para contabilizar todos los mensajes).

2. Análisis de los procesos de "Leer y actualizar datos de los empleados", incluida la posible actualización de los salarios

Suponemos que un usuario deseará primero recuperar y mostrar los datos base del empleado, antes de introducir un cambio en uno o más atributos, incluyendo quizás un nuevo salario. Este procedimiento requerirá dos procesos funcionales. El primero lo desencadena el evento cuando el usuario decide mostrar los datos existentes; Es el proceso funcional de "leer los datos base de los empleados". El segundo es desencadenado por el evento en donde uno o más atributo(s) del empleado han cambiado en el mundo real; es el proceso funcional de "actualizar datos base de los empleados". Los dos procesos funcionales son:

Proceso funcional 'Leer datos de los empleados'. Evento desencadenante: decisión de mostrar los datos existentes

Entrada desencadenante : ID del empleado

Lectura : Datos base del empleado

Lectura : Historial del salario del empleado

Salida : Datos base del empleado

Salida : Historial del salario del empleado

Salida : Mensajes de error/confirmación (en caso de que se ingresara un ID inexistente)

Proceso funcional 'Actualizar datos de los empleados'. Evento desencadenante: los datos de los empleados han cambiado de alguna manera

Entrada desencadenante : Actualizar los datos base de los empleados (para la actualizar uno o más atributo(s))

Entrada : Actualizar salario y su fecha de inicio

Escritura : Datos base del empleado (el registro actualizado)

Escritura : Historial salarial del empleado (se crea un nuevo registro si se ha actualizado el salario)

Salida : Mensajes de error/confirmación (para la entrada de datos inválidos o el posible fallo al actualizar)

3. Análisis del proceso de elaboración del reporte mensual para el departamento de nóminas

Proceso funcional "Reporte mensual de los empleados". Evento desencadenante: Fin del mes

Entrada desencadenante : Señal de fin de mes (cada proceso funcional debe tener una Entrada desencadenante, aunque ésta no transmita datos variables)

Lectura : Datos base del empleado *(para obtener el ID y nombre del empleado)*

Lectura : Historial del salario del empleado (para obtener el salario actual)

Salida : Salario actual del empleado (una línea por cada empleado con su identificación, nombre y salario

Salida : Los totales de empleados al fin de mes (del número de empleados y de su salario total)

NOTAS: La Salida final mueve un grupo de datos que describe el objeto de interés 'Todos los empleados'. No se almacenan datos sobre este objeto de interés, por lo que el grupo de datos es transitorio; pero el objeto de interés es un grupo de personas reales, es decir, una "cosa" real en el mundo del usuario funcional.

No se cuenta un mensaje de error para este proceso funcional ya que no parece haber ninguna razón para que la aplicación tenga que generar tal mensaje. (El sistema operativo puede generar un mensaje de error si no se pueden encontrar los datos, pero esto no forma parte de la aplicación).

EJEMPLO DE SISTEMA EN TIEMPO REAL: Un sistema sencillo de alarma doméstica

Especificación de los requerimientos

Deducimos la funcionalidad disponible para los habitantes normales de la casa y asignada al software, conociendo como se usa el sistema y examinándolo físicamente. No estamos interesados en la funcionalidad proporcionada para el ingeniero de mantenimiento de alarmas, ni en las funciones de configuración del sistema cuando se instala por primera vez.

El principal objetivo del sistema de alarma es activar una o dos sirenas (dispositivos que generan un ruido fuerte) si un sensor detecta un movimiento dentro de la casa o si se abre la puerta frontal.

El software soporta la interfaz humana del sistema de alarma a través de un teclado y LEDs rojos/verdes. El software también acepta datos desde un dispositivo que puede detectar si la puerta frontal de la casa está abierta o no, y de varios detectores de movimiento internos. (El sistema de alarma puede controlar hasta 10 detectores de movimiento. El número no importa para este análisis ya que son todos idénticos y equivalentes). El sistema de alarma también controla una sirena interna y externa.

El sistema de alarma siempre está "encendido", pero no está "activo", es decir, los detectores de movimiento y el sensor de la puerta frontal no funcionan, a menos que un habitante active el sistema. Cuando se activa el sistema, el software se mantiene en un estado donde puede recibir señales de estos sensores, o el software realiza un sondeo de los sensores para obtener su estado. No sabemos qué proceso se utiliza y es irrelevante para medir el tamaño funcional.

Para activar y desactivar el sistema de alarma, el habitante de la casa debe introducir el PIN correcto (número de identificación personal) dentro de un tiempo preestablecido. El PIN es almacenado por el software y se puede cambiar, por lo que debe haber algún almacenamiento persistente. Cuando se introduce el primer dígito de un PIN, se inicia la sirena interna; esta sirena se detiene al ingresar todos los dígitos del PIN correcto. Si se introduce el PIN incorrecto tres veces o si no se introduce el PIN correcto dentro del tiempo preestablecido, también se inicia la sirena externa.

Cuenta con una batería que proporciona continuidad si falla la fuente de alimentación eléctrica de la red, por lo que debe haber un detector de voltaje de energía.

El LED verde se ilumina cuando se conecta a la energía eléctrica. Si se inicia una sirena o si falla la energía eléctrica, se apaga el LED verde y se enciende el LED rojo.

Como ciertas funciones deben completarse dentro de los tiempos preestablecidos, debe haber un mecanismo de reloj. Por ejemplo, si el sistema de alarma se activa antes de salir de la casa, los habitantes deben salir y cerrar la puerta frontal dentro de un número preestablecido de segundos; si no es así, las sirenas se ponen en marcha. La sirena externa no debe continuar por más de un límite legal de 20 minutos.

No sabemos cómo se implementa el reloj, pero por simplicidad, asumimos una implementación de software que se inicia cuando es necesario. La funcionalidad para realizar un seguimiento de los tiempos transcurridos es entonces una forma de manipulación de datos, que podemos ignorar.

Parámetros de la estrategia de medición

Propósito de la medición: Medir los procesos funcionales del software de la aplicación integrada disponible para el habitante de la casa en un funcionamiento normal.

Alcance de la medición: Las funciones del software de la aplicación integrada del sistema de alarma disponibles para el habitante de la casa en un funcionamiento normal. (No estamos interesados si hay un sistema operativo).

Usuarios funcionales: El diagrama de contexto muestra los usuarios funcionales de hardware y cómo interactúan con el software. Tenga en cuenta que los detectores de movimiento son todos funcionalmente idénticos, por lo que no es necesario hacer distinción entre ellos. El usuario humano del sistema de alarma, denominado "el habitante" no es un usuario funcional; él/ella interactúa con la aplicación solamente mediante el teclado y las señales audibles y visuales.

Capa: Aplicación.

Nivel de descomposición: "nivel 0", es decir, sin descomposición.

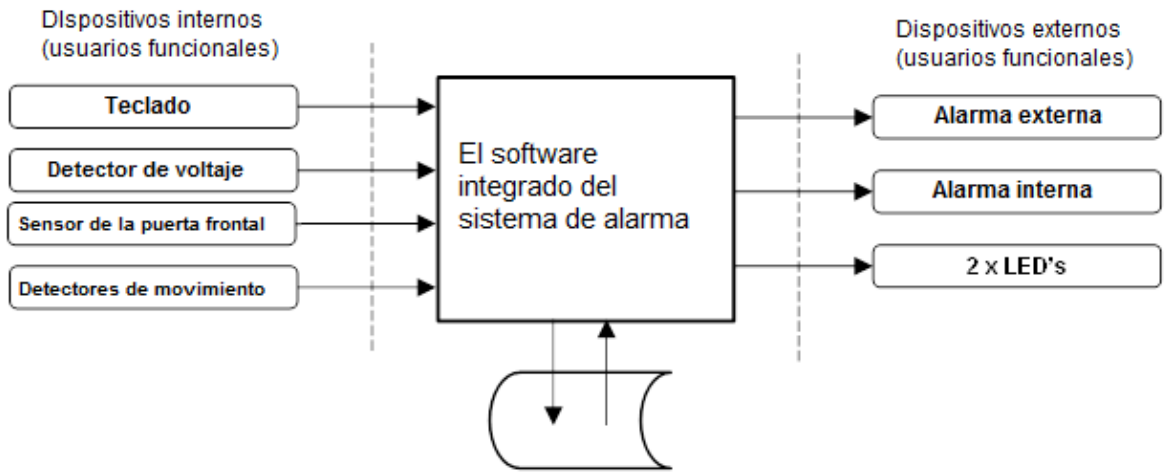


Figura 6.5 – El sistema de alarmas doméstica - Diagrama de Contexto

***Los procesos funcionales**: Después de la configuración inicial, la aplicación del sistema de alarma ofrece al habitante nueve procesos funcionales. Estos pueden identificarse considerando los eventos al que el software debe responder:*

1) *El habitante desea cambiar el PIN existente.*
2) *El habitante desea salir de casa y activar el sistema de alarma.*
3) *El sensor de la puerta frontal detecta que la puerta ha sido abierta y que se activa el sistema de alarma.*
4) *El habitante desea activar el sistema de alarma mientras que él/ella se encuentra en la casa, por ejemplo, por la noche se encuentra fuera del rango de los detectores de movimiento.*
5) *El habitante desea desactivar el sistema de alarma cuando se encuentra dentro de la casa, por ejemplo, al levantarse por la mañana antes de moverse en el rango de los detectores de movimiento.*
6) *Un detector de movimiento indica un movimiento mientras se activa el sistema de alarma (lo que inicia la alarma interna).*
7) *El habitante desea cancelar la(s) sirena(s) y desactivar el sistema de alarma introduciendo el PIN correcto después de los eventos 3) o 6).*
8) *El detector de voltaje señala el fallo de la energía eléctrica de la red.*
9) *El detector de voltaje señala la restauración de los principales suministros de energía eléctrica.*

Análisis de un ejemplo de proceso funcional:

Analizamos el evento 3) en la lista anterior (la puerta frontal se abre mientras el sistema de alarma está activado). Cuando el sensor de la puerta frontal detecta este evento, inicia la sirena interna comienza; se debe introducir el código PIN correcto dentro de un tiempo preestablecido para desactivar el sistema y detener la sirena interna. Si el código PIN no se introduce antes del tiempo preestablecido, o si el código incorrecto se introduce más de tres veces, la sirena externa también se inicia. El proceso funcional tiene los siguientes movimientos de datos.

Proceso funcional: Posible intruso detectado. Evento desencadenante: La puerta se abre mientras está activado el sistema de alarma.

Entrada desencadenante : Mensaje 'puerta abierta' desde el sensor de la puerta frontal

Lectura : Obtener el PIN del almacenamiento persistente

Salida : Mensaje para cambiar el LED verde de “encendido” a “apagado”

Salida : Mensaje para cambiar el LED rojo de “encendido” a “apagado”

Salida : Mensaje para iniciar la alarma interna

Entrada : Código PIN introducido (Si se introduce un código incorrecto, el usuario puede introducir el PIN dos veces más, pero el proceso es siempre el mismo, por lo que sólo se mide una vez).

*- * : Mensaje para cambiar el LED rojo de “encendido” a “apagado” (al ingresar correctamente el PIN.*

*- * : Mensaje para cambiar el LED verde de “apagado” a “encendido” (al ingresar correctamente el PIN)*

Salida : Mensaje para detener la sirena interna (al ingresar correctamente el PIN)

Salida : Mensaje para iniciar la sirena externa (después de tres entradas fallidas del PIN, o si el PIN no se introduce en el tiempo preestablecido)

Salida : Mensaje para detener la sirena externa (después de 20 minutos, un requerimiento legal)

* NOTA: Se trata de repeticiones de las Salidas a los LED anteriores en el proceso, pero con diferentes valores de datos (“encendido” en lugar de “apagado” y viceversa)

6.10 Algunas lecciones generales de estos ejemplos

- Los detalles de la implementación son, por lo general, irrelevantes para el Proceso de Mapeo. Por ejemplo, los procesos funcionales del sistema de personal podrían implementarse de muchas maneras, todos generando los mismos movimientos de datos en el modelo COSMIC. Del mismo modo, no es necesario saber cómo el mensaje ‘puerta abierta’ activa el proceso del sistema de alarma doméstica. (El software, cuando está activo podría sondear el sensor de la puerta frontal y los detectores de movimiento para determinar su estado, o estos sensores podrían enviar su estado al software).
- La secuencia con la que se ejecutaría el sistema, ayuda a entender el proceso para escribir los movimientos de datos de un proceso funcional. Pero la secuencia real será más complicada en la práctica. Por ejemplo, validar los datos ingresados al sistema de personal podría intercalarse con la emisión de ocurrencias de mensajes de error.
- En los ejemplos se muestran la parte de la definición de un proceso funcional que establece que “el conjunto de todos los movimientos de datos de un proceso funcional (que incluye la Entrada activadora) es el conjunto que necesario para cumplir con los FUR para todas las posibles respuestas a su Entrada desencadenante". En el proceso del sistema de personal que actualiza los datos de un empleado, se crea un nuevo historial de salario si se modifica el salario del empleado. En los procesos funcionales del sistema de alarma, los mensajes enviados a los LEDs y/o a las sirenas dependerán, en cada caso, de si el habitante ha introducido el PIN correcto o no. La única tarea del medidor es identificar todos los movimientos de datos necesarios para que un proceso funcional cumpla con los FUR para todas sus posibles respuestas a todos los datos que puede recibir en sus Entradas y Lecturas. El medidor no tiene que preocuparse por la secuencia de los movimientos de datos, ni si son necesarios o no, en cualquier ocurrencia particular del proceso funcional que dependerá de los valores de datos ingresados.
- El conjunto de movimientos de datos de un proceso funcional es el conjunto de tipos, no de ocurrencias.
- La caja del sistema de alarmas es un ejemplo en el que el objeto de interés de cada grupo de datos que entra o sale del software es también el usuario funcional que envía al grupo o que lo recibe (es decir, el usuario funcional envía o recibe datos sobre sí mismo). En estos casos, habiendo identificado a los usuarios funcionales en la fase de Estrategia de Medición, se han identificado igualmente los objetos de interés.

7

EL MÉTODO COSMIC – LA FASE DE MEDICIÓN

Al final de la fase de Mapeo, el Medidor habrá producido un modelo COSMIC de los FUR del software que se va a medir (una instancia del Modelo de Software Genérico). Entonces se puede medir el tamaño funcional de los FUR del software aplicando las reglas de la fase de Medición a este modelo.

7.1 El principio de medición COSMIC

El principio de medición COSMIC refleja el modelo que se muestra en la parte derecha de la figura 6.2.

El principio de medición COSMIC
El tamaño funcional de una parte del software es igual al número de sus movimientos de datos

Un tamaño funcional se mide en unidades 'Puntos de función COSMIC', que se abrevia "CFP". 1 "CFP" se define por convención como el tamaño de un solo movimiento de datos (Entrada, Salida, Lectura o Escritura).

7.2 Composición del tamaño

Los tamaños se pueden medir en varios niveles de composición.

- El tamaño de un proceso funcional es igual al número de sus movimientos de datos
- El tamaño de una parte de software es igual a la suma de los tamaños de sus procesos funcionales
- Es posible determinar el tamaño de una parte del software a partir del tamaño de sus componentes siempre que se sigan las reglas de composición que se dan en el Manual de Medición

La siguiente tabla muestra una forma de registrar los resultados del análisis de los cuatro procesos funcionales del Sistema de personal analizados en la sección 6.9, utilizando la matriz dada en el Apéndice A del Manual de Medición [5].

	Nombres de los grupos de datos							Números de movimientos de datos				
Procesos funcionales del sistema de personal	Datos básicos del empleado	ID del empleado	Historial del salario del empleado	Mensaje de error/confirmación	Fin de mes 'marca de reloj'	Salario actual del empleado	Totales de empleado	Entradas	Salidas	Lecturas	Escrituras	Totales
Crear empleado	E, R, W		E, W	X				2	1	1	2	6
Leer datos del empleado	R, X	E	R, X	X				1	3	2		6
Actualizar datos empleado	E, W		E, W	X				2	1		2	5
Reporte fin de mes	R		R		E	X	X	1	2	2		5
				Totales para el sistema de personal:				6	7	5	4	22

El proceso funcional del sistema de alarmas doméstica analizado en la sección 6.9 tiene 2xEntradas, 1xLectura y 6xSalidas. Su tamaño total es, por lo tanto, 9 CFP.

7.3 Tamaño de los cambios requeridos

El tamaño de algunos cambios requeridos a una parte del software existente, por ejemplo, como el realizado en un proyecto de "mejora", se miden de la siguiente manera:

- El tamaño de un cambio requerido a un movimiento de datos (es decir, que se debe agregar, modificar o eliminar) se mide por convención como 1 CFP. (“Modificado” podría significar cualquier cambio en la manipulación de datos asociada con el movimiento de datos y/o cualquiera de los atributos del grupo de datos movido).
- El tamaño mínimo de un cambio en un proceso funcional es, por lo tanto, 1 CFP.
- El tamaño de todos los cambios requeridos en una parte del software es igual al número de movimientos de datos que deben agregarse, modificarse o eliminarse, sumados a todos los procesos funcionales.

8

VENTAJAS Y BENEFICIOS DEL MÉTODO COSMIC

El método COSMIC para medir el tamaño funcional del software a partir de sus requerimientos es el primer método:

- diseñado de acuerdo con los principios básicos de ingeniería de software,
- en aplicarse al software de aplicaciones de negocios, los dominios de tiempo real y de infraestructura y algunos tipos de software científico/de ingeniería, en cualquier capa de una arquitectura de software, en cualquier nivel de descomposición desde una aplicación completa hasta sus componentes más pequeños,
- para desarrollos de software o para mejoras, independiente de toda la tecnología utilizada y de los métodos de desarrollo,
- diseñado y mantenido por un grupo internacional de expertos en métricas de software,
- diseñado para cumplir con la norma ISO 14143/1 sobre los principios de medición del tamaño funcional,
- completamente abierto y con toda la documentación disponible para su descarga gratuita desde su sitio web www.cosmic-sizing.org.
- y que ha sido aceptada como Norma Internacional (ISO 19761).

En comparación con los métodos de medición del tamaño funcional de '1a generación' (véase el capítulo 3), el método COSMIC:

- es completamente estable debido a sus principios básicos de diseño que no han cambiado desde que el método fue publicado por primera vez. Esto significa que se protege la inversión de una organización en mediciones existentes y que el método será aplicable a futuros paradigmas de software,
- tiene una escala de medida abierta que va de acuerdo con la teoría de la medida. Esto significa que todas las manipulaciones matemáticas de las medidas de tamaño COSMIC son válidas,
- no ha requerido ninguna calibración contra el esfuerzo y, por lo tanto, es una medida pura del tamaño funcional[5].

El método COSMIC es apoyado por:

- documentación exhaustiva; El Manual de Medición ha sido traducido a más de diez idiomas,
- guías que describen cómo aplicar el método a tipos específicos de software, por ejemplo, para almacén de datos (DataWarehouse) o software SOA, o para enfoques específicos de administración de proyectos, por ejemplo, métodos ágiles,
- casos de estudio y herramientas para recolectar e informar los datos medidos,
- datos de referencia completos disponibles en el sitio web www.isbsg.org. Muchas mediciones de proyectos de software en diferentes dominios han demostrado una excelente correlación de tamaños medidos por COSMIC con el esfuerzo del proyecto.

[5] La investigación reciente [15] ha demostrado que los tamaños COSMIC se correlacionan muy bien con los tamaños del mismo software medido por el método de FP Mark II. La escala de tamaño de este último método se calibró relacionando los tamaños con el esfuerzo del proyecto. La buena correlación implica que la escala de tamaño COSMIC debería ser adecuada para los propósitos para los cuales fue diseñada (como un componente para medir el desempeño del proyecto y como parámetro para la estimación del esfuerzo del proyecto), aunque el método del tamaño no fue calibrado contra el esfuerzo.

- servicios de proveedores, proveedores de capacitación, consultoría, herramientas de estimación, etc.,
- un examen de certificación de nivel de ingreso,
- guías para asegurar la exactitud y la comparabilidad de las mediciones,
- variantes documentadas para el dimensionamiento aproximado de COSMIC que pueden utilizarse en fases tempranas de la vida de un proyecto cuando aún no se han establecido todos los detalles para los requerimientos o para medir de forma rápida el tamaño [7],
- métodos documentados de conversión de tamaños medidos mediante métodos FSM de 1a. generación a tamaños COSMIC utilizando correlaciones estadísticas [16],
- grupos de usuarios activos en Linkedin ('Usuarios COSMIC') y Twitter (@COSMIC_FSM),
- su sitio web www.cosmic-sizing.org donde existe un foro para hacer preguntas y para analizar, así como para dar noticias.

El método COSMIC se utiliza con éxito en todo el mundo para medir el desempeño del proyecto, para estimar, el control del alcance del proyecto, etc. El siguiente mapa mental muestra una gama de usos posibles de las mediciones del tamaño funciona.

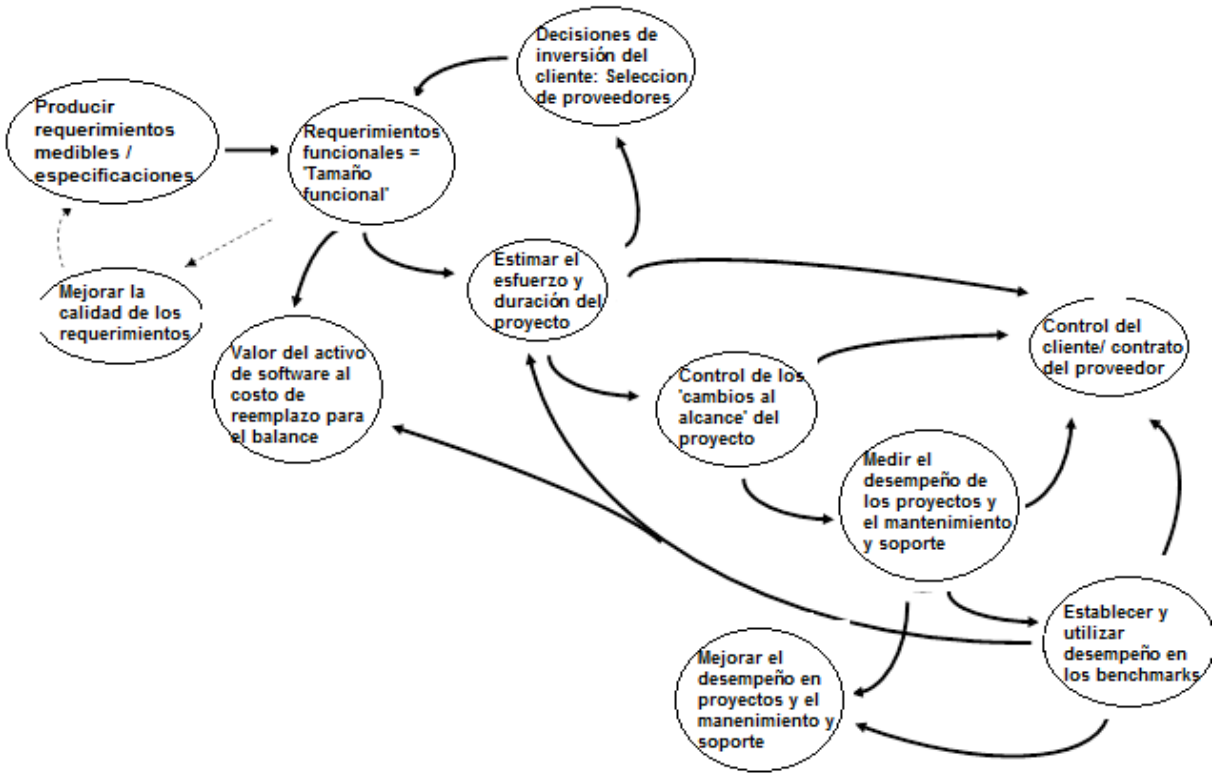


Figura 8.1 – Mapa mental de posibles usos de medidas de tamaño funcional

El método COSMIC también está siendo ampliamente estudiado por la comunidad de investigación académica. Entre estos se destacan varios enfoques para automatizar la medición del tamaño de COSMIC a partir de, por ejemplo, Requisitos en UML y de ejecutar programas.

El sitio web www.cosmic-sizing.org contiene un gran número de trabajos de investigación y conferencias.

REFERENCIAS

[1] ISO 19761:2011 – Software Engineering – COSMIC: a functional size measurement method' obtainable from www.iso.org.

[2] The COCOMO II Estimating Model', www.csse.usc.edu/csse/research/COCOMOII

[3] Albrecht, A.J., 'Measuring Application Development Productivity', IBM Applications Development Symposium, Monterey, October 1979

[4] ISO 14143-1:2012 – Software and Systems Engineering – Software measurement – Functional size measurement – Definition of concepts

[5] The 'COSMIC Functional Size Measurement Method, version 4.0.1: Measurement Manual'. (The COSMIC Implementation Guide for ISO 19761: 2011), April 2015

[6] Guideline for 'Measurement Strategy Patterns': Ensuring that COSMIC size measurements may be compared, version 1.0, March 2013

[7] Guideline for early or rapid COSMIC functional size measurement by approximate approaches', July 2015

[8] Guideline for sizing Business Application software, version 1.1, May 2008

[9] Guideline for sizing Real-time software, version 1.0, June 2012

[10] Guideline for sizing Data Warehousing application software, version 1.0, May 2009

[11] Guideline for sizing Service-Oriented Architecture software, version 1.0, April 2010

[12] Guideline for the use of COSMIC FSM to manage Agile projects, version 1.0, September 2011

[13] Guideline for mapping from the Unified Modeling Language to COSMIC FSM (in preparation)

[14] Guideline on Non-functional and Project requirements: how to consider non-functional and project requirements in software project performance measurement, benchmarking and estimating, v1, November 2015

[15] A, Dasgupta, C. Gencel, C., Symons, 'A process to improve the accuracy of MkII FP to COSMIC size conversions: insights into the COSMIC method design assumptions', IWSM 2015, Krakow, Poland.

[16] 'Guideline on how to convert 'First Generation' Function Point sizes to COSMIC sizes' (to be published early 2016)

APÉNDICE A. CAMBIOS PRINCIPALES DE LA V4.0 A LA V4.0.1

V4.0 Ref.	Cambio
General	Las referencias al sitio web www.cosmic-sizing.org sustituyen las referencias al sitio www.cosmic-**sizing**.org
General	Las referencias a otras publicaciones de COSMIC han sido actualizadas a las existentes a fines de 2015.
2.6	La definición COSMIC de requerimientos no funcionales (NFR) ya no incluye los requerimientos y restricciones del proyecto. Estos son ahora definidos y tratados por separado como NFR en [14]. Este cambio ha dado lugar a cambios menores en los ejemplos de la sección 2.6.
6.9	La descripción en la Fase de Mapeo de las suposiciones sobre el ejemplo simple del sistema de personal ha tenido cambios editoriales menores para mejorar la claridad.
6.9	El ejemplo en tiempo real de un sistema de alarma doméstica necesitaba varias aclaraciones y correcciones, las más importantes de las cuales fueron las siguientes: • El texto confunde el “sistema de alarma” con la “alarma” (lo que hace ruido fuerte). Este último fue re-nombrado “sirena”. • Algunos aspectos de la descripción de la funcionalidad del sistema de alarma se reescribieron para hacerlos más claros y más completos. La lista de eventos fue reescrita para aclarar al usuario funcional lo que causa o se percibe del evento. • Se reconoció la necesidad de un proceso funcional adicional (número 7).
8	Se han añadido resultados de investigaciones recientes sobre la conversión de la medición del tamaño funcional (Nota de pie de página 5) y sobre la automatización de la medida del tamaño COSMIC, lo que indica otras ventajas potenciales del método COSMIC.

APÉNDICE B - COSMIC SOLICITUD DE CAMBIOS Y PROCEDIMIENTO PARA COMENTARIOS DEL MÉTODO COSMIC

El Comité de Prácticas de Medición COSMIC (MPC, por sus siglas en inglés) está deseoso por recibir retroalimentación, comentarios y, si es necesario, solicitudes de cambio para el Manual de Medición COSMIC. Este apéndice establece cómo comunicarse con el MPC COSMIC.

Todas las comunicaciones al MPC COSMIC deben enviarse por correo electrónico a la siguiente dirección:

mpc-chair@cosmic-sizing.org

Comentarios y Sugerencias Generales Informales

Se enviarán por correo electrónico a la dirección mencionada las retroalimentaciones y/o sugerencias informales sobre el Manual de Medición, así como comentarios sobre cualquier dificultad para entender o aplicar el método COSMIC, sugerencias de mejora general, etc. Los mensajes se registrarán y se confirman, por lo general, dentro de las dos semanas de su recepción. El MPC no puede garantizar la acción ante tales comentarios generales.

Solicitudes de cambio formal

Cuando el lector del Manual de Medición considera que hay un error en el texto, una necesidad de aclaración o que algún texto necesita mejorar, se puede presentar una solicitud de cambio formal (CR). Los CR formales se registrarán y se confirmarán dentro de las dos semanas siguientes a su recepción. A cada CR se le asignará un número de serie y se distribuirá a los miembros del MPC COSMIC, un grupo mundial de expertos en el método COSMIC. Su ciclo de revisión normal toma un mínimo de un mes y puede tomar más tiempo si el CR resulta difícil de resolver. El resultado de la revisión puede ser que, el CR será aceptado, rechazado o "colocado como pendiente de análisis adicional" (en este último caso, por ejemplo. si hay una dependencia de otro CR), el resultado se comunicará de nuevo al solicitante tan pronto como sea posible.

Un CR formal sólo será aceptado si está documentado con toda la información siguiente:

- Nombre, posición y organización de la persona que solicita el CR
- Detalles del contacto que solicita el CR
- Fecha de presentación
- Enunciado general del propósito del CR (p. ej., “necesitad de mejorar el texto…”)
- Texto real que necesita ser cambiado, reemplazado o borrado (o referencia clara al mismo)
- Propuestas de texto adicional o de reemplazo
- Explicación completa del por qué es necesario el cambio

En el sitio web www.cosmic-sizing.org se encuentra disponible un formato de solicitud para enviar un CR.

La decisión del MPC COSMIC sobre el resultado de una revisión CR y, si se acepta, en qué versión del Manual de Medición se aplicará el CR, es concluyente.

Preguntas sobre la aplicación del método COSMIC

El MPC COSMIC lamenta no poder responder preguntas relacionadas con el uso o aplicación del método COSMIC. Existen organizaciones comerciales que pueden proporcionar formación, asesoría o apoyo de las herramientas para el método. Por favor consulte el sitio web www.cosmic-sizing.org para más detalles.