UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

U.A.C.P. y P.

I.I.M.A.S.

ANÁLISIS DE PUNTOS DE FUNCIÓN

T E S I S

QUE PARA OBTENER EL GRADO DE:
MAESTRA EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:
CECILIA PÉREZ COLIN

BIBLIOTECA DEL
I. I. M. A. S.

DIRECTOR DE TESIS:
DRA. HANNA OKTABA

MEXICO, D.F. 1999

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

ÍNDICE

ANÁLISIS DE PUNTOS DE FUNCIÓN

INTRODUCCIÓN

En la industria del desarrollo de sotware, siempre ha habido la necesidad de saber cómo conocer el tamaño de un programa con respecto a su funcionalidad, independiente de la técnica desarrollada en su construcción y de las decisiones de implementación. Poder medir un sistema de esta manera significaría, en términos generales, poder calificar la calidad de un producto de software.

Al medir un sistema se obtiene información, entre otras cosas, sobre la estimación del tiempo necesario para la construcción de un sistema; el costo dependiendo de la funcionalidad, la productividad del equipo de trabajo o de la organización, comparar diferentes técnicas y tecnologías, etc.

Existen diversas métricas para este fin. De las más conocidas están la de contar el número de líneas de código en un sistema, ver el número de bytes que ocupa un programa en particular o determinar el tiempo necesario para el desarrollo del sistema. Al medir los sistemas con este tipo de métricas, en realidad no se mide la complejidad y funcionalidad del sistema, pues dependen de factores como el lenguaje de programación, el ambiente con el que se está trabajando e inclusive las personas que trabajan en el sistema.

La metodología para medir el tamaño de los programas que se explicará en esta tesis, intenta eliminar los factores de dependencia de las métricas anteriores.

El **análisis de puntos de función**, es otra opción de medida de tamaño de gran significado. Esta se hizo pública por primera vez por Allan Albrecht de IBM en 1979. Este metodología cuantifica las funciones contenidas dentro del software en términos significativos para los usuarios del mismo.

Punto de función es una de las métricas más reconocidas en la ingeniería de software y, de hecho, es casi ya la unidad de medida estándar para conocer la complejidad de los programas en muchos países. Existe una asociación internacional de usuarios denominada IFPUG (International Function Point Users Group) encargada de unificar los criterios de medición con esta metodología y promover su uso y evolución.

El objetivo de esta tesis es estudiar y describir en términos sencillos, cómo medir qué tan complejo es el software mediante esta métrica, dar ejemplos con casos de estudio y medirlos con puntos de función para clarificar este concepto. Estudiar además cómo diversos usuarios de esta tecnología han adaptado esta métrica a tecnologías más nuevas como son las aplicaciones en Internet y las aplicaciones con interfaz gráfica de usuario que son ahora más comunes.

Esta tesis consta de 5 capítulos, en los cuales se describe la teoría y aplicación del análisis de puntos de función.

El capítulo 1 describe el por qué medir en la ingeniería de software, las métricas más utilizadas en esta área, una breve historia del Análisis de Puntos de Función y el por qué de su creación.

El capítulo 2 define los conceptos necesarios para el Análisis de Puntos de Función, indicando las reglas y estableciendo ejemplos que permiten aclarar cada uno de ellos.

El capítulo 3 describe la metodología indicando los pasos a seguir en el proceso de medición de puntos de función.

El capítulo 4 ejemplifica la metodología con una aplicación práctica.

El capítulo 5 muestra un panorama de las adaptaciones que han hecho algunos usuarios de puntos de función a nuevas tecnologías de desarrollo de software.

El capítulo 6 indica conclusiones y comentarios sobre el uso de esta metodología.

Se concluye con un anexo que contiene formas sugeridas de ayuda para llevar a cabo el análisis, un glosario de términos y la bibliografía.

CAPÍTULO I
ANTECEDENTES

1.1 INTRODUCCIÓN A LAS MÉTRICAS

Medir es vital para el desarrollo de la ciencia. Las siguientes citas confirman su utilidad.

Cuando puedes medir y expresar en números lo que estas diciendo, es porque conoces algo acerca de eso; pero cuando no puedes medirlo ni expresarlo en números, tienes un conocimiento pobre e insatisfactorio: Este puede ser el principio del conocimiento, pero apenas existe en tu pensamiento avances de la ciencia. (Lord Kevin, 1824-1904)

No se puede controlar lo que no se puede medir. (DeMarco, 1982)

Este punto de vista para aplicar la medición dentro de la ciencia también es sustentado por científicos que se dedican a escribir acerca de teorías de medición, Fred S. Roberts expresa lo siguiente acerca de la teoría de medición:

"La mayor diferencia entre una ciencia 'bien desarrollada' como la física y otras ciencias 'menos desarrolladas' como la psicología o la sociología es el grado en que las cosas que las conforman son medidas."

Sin embargo, no sólo las ciencias hacen uso de las mediciones. En la vida real utilizamos medidas para hacer los conceptos más visibles y por lo mismo más entendibles y controlables. Por ejemplo, medir la distancia nos indica cuánto tiempo resta para alcanzar un destino cuando viajamos, medir la temperatura ambiente nos indica si debemos salir abrigados o no, etc.

La siguiente sección muestra la utilidad de medir en la ingeniería de software, pero antes es necesario definir lo que es una métrica de software.

Una *métrica de software* es cualquier medición que se relaciona con el producto o proceso de software.

1.2 MEDIR EN LA INGENIERÍA DE SOFTWARE

La ingeniería de software describe un conjunto de técnicas que aplican un enfoque de ingeniería a la construcción y soporte de productos de software. Por enfoque de ingeniería se entiende que cada actividad es comprendida y controlada de tal forma que hay pocas sorpresas conforme el software se especifica, diseña, construye y mantiene. Sin embargo, una gran cantidad de proyectos de ingeniería de software caen en excesos de presupuesto y tiempos de desarrollo.

Medir el tamaño del software es necesario para tomar decisiones sobre costo, productividad, calidad y reuso del software, cambios en el proceso y en la metodología con la que se construyó, etc.

Costo.- El costo es un tema importante para las empresas que se dedican a la construcción de software a la medida. Muchas empresas calculan los costos de sus sistemas al finalizar la fase de requerimientos. Este cálculo lo hacen presuponiendo la duración proyectada y las horas hombres que supuestamente van a necesitar en el transcurso del desarrollo de la aplicación. Sin embargo, por lo general la duración difiere mucho de lo que realmente se requiere para el desarrollo completo del sistema y por lo tanto, el costo final rara vez es el justo.

Es por eso que es necesario contar con información precisa de qué tan complejo es el software en las fases más tempranas del desarrollo y poder tomar así decisiones importantes como: el costo justo por el desarrollo, el tiempo necesario para su construcción, si es conveniente hacer un sistema en vez de comprarlo o reemplazar un sistema en vez de mejorarlo.

Productividad.- Saber qué tan productiva es una empresa, un grupo de trabajo, un participante en particular, puede ayudar a alentar el trabajo desempeñado. Por ejemplo, es posible tener información del incremento en la productividad de la empresa en un cierto periodo de tiempo, cuál es el mejor grupo de trabajo, quién es el mejor participante en alguna de las actividades (un analista, un programador, un ensayista, etc.)

Calidad y reusabilidad del software.- Las métricas en estos casos se usan para demostrar los niveles existentes de calidad y para monitorear los esfuerzos continuos de mejoramiento en el proceso de desarrollo de software. No hay forma de decir que un proyecto se está haciendo correctamente sin que se tenga una medida de lo correcto de un programa. Es esencial que se midan y registren las características tanto de los buenos proyectos como de los malos.

Una buena medida de calidad es poder determinar la proporción de defectos o fallas existentes en un sistema con respecto a su tamaño funcional o por tiempo. Otro indicador de calidad importante es medir las horas de mantenimiento por tamaño del sistema, en la que se de una relación entre el esfuerzo de soporte y el tamaño del sistema ya instalado. Si esta relación es muy grande, podría pensarse en hacer una reingeniería nueva o hasta reemplazar el sistema.

Cambios en el proceso y metodología.- Poder medir una técnica ayuda a saber si existe un buen desempeño en los productos o servicios desde el punto de vista del desarrollador. Las mediciones técnicas pueden ser utilizadas para lograr un análisis más eficiente y por ejemplo, mejorar el desempeño en el diseño. Por otro lado, también es necesario medir la funcionalidad de un programa para medir el desempeño en los productos o servicios desde la perspectiva del usuario. Medir la funcionalidad en una aplicación debe ser independiente de la técnica y de la implementación elegida. Se pueden comparar técnicas y metodologías utilizadas y poder elegir la más productiva.

En conclusión, medir nos ayuda a tomar mejores decisiones.

Jones en [11] menciona que las empresas que más éxito han tenido en tener bajo control su software tienen las siguientes características:

1. Miden con precisión la productividad y calidad del software.
2. Planean y estiman con precisión los proyectos de software.
3. Tienen grupos de trabajo, con administradores y técnicos capaces.
4. Tienen una buena estructura de organización.
5. Tienen métodos y herramientas de software efectivos.
6. Tienen un ambiente de trabajo adecuado.
7. Son capaces de reusar grandes cantidades del software que han producido.

1.3 OTRAS MÉTRICAS DE TAMAÑO DE SOFTWARE

Dada la importancia de tener una métrica para poder estimar el tamaño del software y poder tener control sobre éste, se han propuesto varias, entre las que destacan el tiempo total invertido, conteo de líneas de código, la métrica de Halstead y el análisis de puntos de función.

1.3.1 Tiempo total invertido

El tiempo total invertido son las horas necesarias para completar un proyecto. Este métrica no mide lo que se produce sino el esfuerzo necesario para producir un proyecto. Aquí el cliente no controla la habilidad, efectividad o eficiencia del programador. El cliente paga por el número de horas necesarias para completar un proyecto, incluyendo el costo por pruebas.

1.3.2 Líneas de código

Una de las técnicas para medir software más comúnmente utilizada es contar las líneas de código que contiene un sistema. Los problemas con esta técnica son muchos, sin embargo, la más importante es que no mide la funcionalidad de producto y depende mucho del lenguaje o metodología utilizada para su construcción. Con esta métrica resulta que entre más poderoso y avanzado sea el lenguaje de programación, parece menos productivo que los lenguajes llamados de bajo nivel, para esta métrica nunca hubo un estándar que diera cierto peso a los lenguajes. Un sistema que cuente con 3 mil líneas de código podría ser mucho más rico en funcionalidad que otro de 5 mil líneas o si dos programas tienen la misma funcionalidad pero programados en diferente lenguaje o metodología, pueden diferir mucho en el número de líneas de código escritas. Otro de sus inconvenientes es que solamente se puede medir software ya construido y no es posible hacerlo desde las primeras fases de desarrollo.

1.3.3 La métrica de Halstead

La métrica del Dr. Maurice Halstead de la Universidad de Purdue fue desarrollada en 1977 para medir la complejidad de los módulos de un programa, esta métrica divide las líneas de código en dos unidades atómicas: la porción ejecutable (a la que determinó "operadores") y la porción que describe los datos (que determinó "operandos"). Esta métrica cuenta cuatro valores por separado:

1. El número total de operadores únicos.
2. El número total de operandos únicos.

3. La cantidad total de operadores en una aplicación.

4. La cantidad total de operandos en una aplicación.

De estos cuatro resultados, se calcula lo que llamó:

1. El vocabulario del programa (la suma de los operandos y operadores únicos).

2. La longitud del programa (la suma total de los operadores y los operandos).

Aunque no es tan subjetiva como la de contar líneas de código, tampoco puede decirse que mida la funcionalidad de un sistema desde el punto de vista del usuario. Otro inconveniente es que no puede medir el sistema en las primeras fases del desarrollo.

1.3.4 Análisis de puntos de función

El análisis de puntos de función es un metodología utilizada para medir la complejidad y eficiencia del software, desde el punto de vista del usuario, es decir, cuenta solamente lo que el usuario pide que tenga el sistema y lo que recibe a cambio. Es importante mencionar que el usuario al que nos referiremos es el que entiende el sistema desde una perspectiva funcional a diferencia del usuario final que es el que va a ser uso de éste sin importarle como es que funciona.

Se basa en inspeccionar la aplicación y determinar su funcionalidad. Entre las ventajas que proporciona está la de ser independiente a cualquier lenguaje y tecnología además de poderse aplicar en cualquier etapa del ciclo de vida del desarrollo del proyecto, desde la definición de requerimientos hasta el mantenimiento de la aplicación. Es una metodología que cuenta con estándares bien definidos y por lo tanto el resultado es confiable, en términos de que se llegaría al mismo resultado sin importar quién la aplique.

A grandes rasgos, esta metodología se basa en lo siguiente: usando un conjunto básico de criterios, cada una de las funciones del negocio en una aplicación se representa por un número de acuerdo a su tipo y complejidad. Estos criterios se basan en el número de archivos lógicos internos (ILF), archivos de interfaz externa (EIF), entradas (inputs), salidas (outputs), consultas (inquiries), contenidos dentro de cada una de las funciones del negocio. La suma de todos estos números da una medida inicial que se ajusta después incorporando un número de factores relacionados al software como un todo. El resultado final se da en términos de "número de funciones" que mide el tamaño y complejidad del producto de software.

Esta metodología es la que se explicará en los capítulos siguientes.

1.4 HISTORIA DE LOS PUNTOS DE FUNCIÓN

El concepto de puntos de función fue introducido por primera vez por Allan Albrecht en su exposición "Measuring Application Development Productivity" dentro de las conferencias publicadas en *Procceding of the Joint IBM/Share/Guide Application Development* en 1979. Este primer modelo es un subconjunto del que actualmente se utiliza, pues a partir de la definición de la técnica, se ha ido modificando y mejorando hasta llegar a una versión estandarizada. Es importante mencionar que en los años siguientes a la introducción de Puntos de Función, el tema de Métricas del Software ha surgido con mayor intensidad y se imparten varias conferencias en congresos dedicados a este tema.

Dado el éxito de esta métrica, en 1986 se estableció el IFPUG, que son las iniciales de "International Function Points Users Group" (http://www.ifpug.org), grupo internacional creado con el fin de promover y alentar el uso de los puntos de función y cuya base se encuentra en Westerville, Ohio. Entre los principales beneficios que ha traído la conformación de este grupo han sido el de reglamentar y estandarizar los criterios en los que se basa el conteo de puntos de función y el de reunir a varios usuarios de puntos de función con el fin de discutir y exponer casos de estudio que sirvan como ejemplos a otros.

Con el paso de los años, se ha ido modificando la descripción inicial. En enero de 1994, IFPUG publica la última versión hasta ahora del "Manual de Prácticas de Conteo de Puntos de Función", en su versión 4.0, y es en base a ésta que se describe con detalle la metodología en el capítulo 2.

Hasta 1996 se han publicado siete versiones diferentes sobre puntos de función. Las primeras tres cambiaron un poco la estructura del análisis de puntos de función, mientras que las cuatro últimas son versiones de IFPUG enfocadas a clarificar las reglas y dar una guía para medir. La versión 4.1 está lista y se publicará en el transcurso de 1999.

Otra de las funciones interesantes de IFPUG es la proporcionar cursos para formar lo que llaman "Analista de Puntos de Función". En 1991 abre un programa de certificación que se basa en el entrenamiento de personas sobre esta metodología. Este mismo año, IFPUG abre un programa que otorga el certificado "Especialista en Puntos de Función".

En IFPUG se han creado desde 1988 una serie de comités y grupos de trabajo que se encargan de diversas tareas necesarias para su expansión, los principales son:

- Comité de prácticas de conteo - CPC (Counting Practices Committee) encargado de publicar los estándares de cómo deben definirse y contarse los puntos de función.
- Comité del curriculum de educación - ECC (Education Curriculum Committee) formado en 1992 y responsable de diversos aspectos de educación y entrenamiento en el dominio de esta métrica. incluyendo conferencias, congresos y los cursos de certificación.
- Comité de certificación - CC (Certification Committee) formado en 1991 y encargado de diseñar los exámenes de certificación.
- Comité de reporte de administración - MRC (Management Reporting Committee) tiene que ver con indicar cómo deben ser utilizados los puntos de función para medir productividad y calidad.
- Comité de nuevos ambientes - NEC (New Environments Committee) se encarga de estudiar la adaptación de puntos de función para nuevos ambientes de programación.

Con lo anterior, nos damos cuenta de la importancia que ha llegado a adquirir esta técnica. A la fecha, IFPUG tiene 1200 miembros en más de 30 países y existen diversas organizaciones en diversos países afiliados a IFPUG, por ejemplo: en Europa existe EAFPUG, iniciales de "European Association of Function Points Users Group".

1.5 ESTANDARIZACIÓN DE LOS PUNTOS DE FUNCIÓN

La idea del grupo IFPUG es que *Analysis Function Points* sea la métrica más reconocida de tamaño funcional del software a nivel internacional que cumple con la norma de ISO/IEC 14143.

ISO (International Organization for Standarization) e IEC (International Electrotechnical Comission) forman un grupo especializado para hacer estándares a nivel mundial a través de comités técnicos establecidos por distintas organizaciones que tienen interés en estandarizar cierta tecnología en particular.

ISO/IEC 14143 que se conoce bajo el título *de Tecnología de la información - Medición del Software - Medición del tamaño funcional*, agrupa las características y requerimientos que deben tener los métodos para medir el tamaño del software en términos de funciones requeridas por el usuario y consta de 5 partes:

Parte 1.- Definición de conceptos.

Parte 2.- Aseguramiento de conformidad de los métodos para medir el tamaño del software con respecto a ISO/IEC 14143-1.

Parte 3.- Verificación de los métodos para medir el tamaño del software.

Parte 4.- Medición del tamaño funcional. Modelo de referencia.

Parte 5.- Determinación de los dominios funcionales para utilizar con la medición del tamaño funcional.

Entre las características más importantes que exige esta norma está la de que la métrica debe basarse en los requerimientos de funcionalidad del usuario, vistos desde su perspectiva, debe ser independiente a los métodos, técnicas, componentes físicos, etc. utilizados para el desarrollo de la aplicación a medirse. Uno de los requerimientos principales es el que debe estar bien definida la unidad con la cual se expresa el tamaño funcional, así como describir el tipo de información necesaria que permite aplicar el método.

El Análisis de Puntos de Función es la primera metodología publicada que abarca el concepto de medir el tamaño funcional, sin embargo, eso no quiere decir que sea la única que cumple con este estándar. Existen variantes de puntos de función que bien cumplen con las características y requerimientos solicitados por ISO/IEC.

En México, este estándar internacional ISO/IEC 14143 se encuentra en proyecto de adaptarse como norma mexicana. Con esta norma se podría determinar la métrica más apropiada que mida los proyectos de software desarrollados en México. En nuestro ámbito, los puntos de función podrían llegar a ser la métrica que mida la funcionalidad y eficiencia del software más adecuada. Esto traería muchas ventajas para la industria mexicana, ya que los contratos se medirían más que por el producto en sí, por la métrica estándar. En caso que se decidiera que los puntos de Función fueran esta métrica estándar, sería necesario que todas las empresas dedicadas a la ingeniería de software adoptáran esta metodología y prepararan gente especializada en medir en base a puntos de función. Actualmente en México existen pocas personas con la certificación de especialistas otorgada por IFPUG, de hecho, la primera persona en Latinoamérica que cuenta con esta certificación fue calificada como tal en 1996.[1]

[1] Información proporcionada por el Grupo Certum, empresa mexicana impulsora de este proyecto.

CAPÍTULO II
CONCEPTOS PARA EL ANÁLISIS DE PUNTOS DE FUNCIÓN

Este capítulo y el siguiente describe la metodología del análisis de puntos de función. Proporciona la definición, objetivos e ideas generales del análisis, y el beneficio de su uso. Define los componentes necesarios para el cálculo junto con ejemplos comunes para poder identificarlos en una aplicación a medir. Con los conceptos ya identificados, el capítulo tres explica los pasos necesarios para calcular el tamaño de un sistema con esta metodología.

2.1 OBJETIVOS E IDEAS GENERALES EN CUANTO AL ANÁLISIS DE PUNTOS DE FUNCIÓN

El análisis de puntos de función es un metodología estándar para medir el desarrollo y la productividad del software desde el punto de vista del usuario, independiente del lenguaje de programación, método de desarrollo y tecnología utilizada para su implementación. Se basa en la inspección de la aplicación y mide el tamaño del sistema en términos de números de funciones. En el análisis de puntos de función se divide y clasifica componentes funcionales de un sistema a fin de entenderlo y analizarlo mejor.

Esta medida es consistente entre varios proyectos y organizaciones gracias a la estandarización que ya existe. Uno de los objetivos principales al utilizar esta métrica es que el resultado del conteo del tamaño de un cierto sistema sea el mismo sin importar quién haya realizado la medición.

El proceso de evaluar el tamaño del software no debe representar una carga excesiva en el proceso global de generación del mismo, puesto que eso repercutiría en el tiempo de desarrollo del software y los costos correspondientes. Con la experiencia que vaya adquiriendo un analista de puntos de función, esta carga se minimiza. Además el cálculo de puntos de función puede hacerse en diferentes etapas del desarrollo de una aplicación; desde la fase de requerimientos hasta la fase de mantenimiento.

Entre los principales beneficios que se obtiene al hacer un análisis de puntos de función están los siguientes:

- Una herramienta para medir paquetes de aplicación adquiridos.

- Una herramienta para determinar el beneficio de una aplicación para una organización contando funciones que coinciden de manera específica con los requerimientos solicitados.
- Una herramienta para medir las unidades de un producto y obtener otras medidas de productividad.
- Un vehículo para estimar costos y recursos necesarios para el desarrollo y mantenimiento del software (también se deben considerar atributos del proyecto, estructura de repartición de trabajo, etc.)
- Un factor de normalización para comparar productos de software.

2.2 COMPONENTES PARA LA APLICACIÓN DE LA METODOLOGÍA

Para poder llevar a cabo el proceso de medir con puntos de función, es necesario tener toda la documentación de la aplicación disponible. Por ejemplo, una breve descripción de la aplicación desde el punto de vista del usuario, diagramas que muestren la interacción con otras aplicaciones, requerimientos del usuario, diagramas de flujo, la estructura de la base de datos, el código fuente de la aplicación, formatos de pantallas, reportes, etc. También es recomendable tener contacto frecuente con el equipo que desarrolla el sistema.

Para entender la metodología del análisis de puntos de función es necesario primero definir los principales conceptos y componentes de una aplicación por medir, que son los siguientes:

- **El tipo de conteo**
- **La frontera de la aplicación**
- **La funcionalidad de datos que son:**
  Archivos lógicos internos - ILF (Internal Logical Files)
  Archivos de interfaz externa - EIF (External Interface Files)
- **La funcionalidad de transacciones que son:**
  Entradas externas - EI (External Inputs)
  Salidas externas - EO (External Outputs)
  Consultas externas - EQ (External Inquiry)

- **Para cada una de las funcionalidades anteriores, identificar los elementos con los que vamos a determinar la complejidad.**
  Elementos de datos - DET (Data Element Types)
  Archivos referenciados - FTR (File Types Referenced)
  Elementos de registro - RET (Record Element Types)
- **Las 14 características generales del sistema que son:**.

1. Comunicación entre datos.
2. Procesamiento distribuido de datos.
3. Objetivos de desempeño.
4. Grado de configurabilidad.
5. Volumen de las transacciones.
6. Captura de datos en-línea (interactiva).
7. Eficiencia para el usuario final.
8. Actualización de datos en-línea.
9. Complejidad del proceso.
10. Reusabilidad.
11. Facilidad de instalación y conversión.
12. Facilidad de operación.
13. Posibilidad de instalación múltiple.
14. Facilidad de modificación (mantenible).

A continuación se dan las definiciones y ejemplos de estos componentes. Las definiciones y reglas son en base al manual de conteo de IFPUG [4].

2.2.1 Tipos de conteo

Existen 3 tipos de conteo de puntos de función con respecto al tipo de proyecto que vamos a medir.

- **Conteo de proyectos por primera vez**. Mide las funciones proporcionadas a los usuarios en una primera medición de la aplicación. Puede ser desde la fase de requerimientos hasta la primera instalación del software cuando el proyecto está concluido.

- **Conteo por mantenimiento del proyecto**. Mide sólo las modificaciones por cambios en la aplicación dadas al usuario al agregar, cambiar o eliminar funcionalidad en el software ya existente. Este tipo de medición no nos proporciona el tamaño total del sistema, solamente refleja los cambios en las funcionalidades.
- **Conteo actualizado del proyecto**. Es la última funcionalidad proporcionada al usuario por la aplicación. Este conteo cambia cada vez que se hacen modificaciones en la aplicación.

Ejemplo: La compañía "SINT" dedicada a la construcción de software solicitó hacer un primer conteo del proyecto "X" para saber el tamaño del software entregado a su cliente (en este caso se realiza un "conteo de proyecto por primera vez"). Sin embargo, tres meses después se hicieron cambios que afectaron su funcionalidad y ahora la compañía desea saber el tamaño de esos cambios efectuados (en este caso se realiza un "conteo por mantenimiento del proyecto" al mismo proyecto "X"). Ya con esos cambios, "SINT" le interesa saber finalmente qué tamaño tiene el proyecto "X" (se realiza el "conteo actualizado del proyecto").

2.2.2 Frontera de la aplicación

Para poder identificar los componentes de los puntos de función. Debemos distinguir entre la aplicación que estamos midiendo y las aplicaciones externas o el mismo dominio del usuario que de alguna manera interactúan con nuestra aplicación. Debemos tener clara la diferencia entre la información que de alguna manera nuestro sistema está manteniendo (donde mantener significa que podemos añadir, actualizar y/o eliminar ciertos datos) y la información a la que hacemos referencia pero que otro sistema se encarga de mantener. Esta frontera se define desde el punto de vista del usuario.

Para que se considere la frontera de la aplicación, se deben cumplir las siguientes reglas:

- La frontera se determina desde el punto de vista del usuario. El enfoque está en lo que el usuario puede entender y describir.
- La frontera de las aplicaciones que se relacionan se basan en las separaciones del negocio vistas por el usuario y no en los intereses tecnológicos.
- Para los proyectos en mantenimiento la frontera debe coincidir con la frontera ya establecida en el conteo de proyectos por primera vez.

Ejemplo. Suponga un sistema sencillo de conversión de moneda, que al introducir una cierta cantidad en pesos (moneda nacional) y elegir la moneda de otro país, regresa el equivalente de esa cantidad en la moneda extranjera elegida. Suponga además que la aplicación interactúa con el banco, para proporcionar el tipo de cambio actual de las monedas extranjeras. En este caso el sistema de conversión de moneda es la aplicación a medir, por lo que se encuentra dentro de la frontera de la aplicación, y el sistema bancario encargado de actualizar los tipos de cambio, que es un sistema que el sistema de conversión no puede mantener, se encuentra fuera de la frontera de nuestra aplicación.

2.2.3 Funcionalidad de datos

Representan la funcionalidad dada al usuario por grupos de datos mantenidos dentro de la aplicación o leídos desde otras aplicaciones. Estas se componen de los siguientes tipos de archivos: Archivos lógicos internos (ILF) y Archivos de interfaz externa (EIF).

Archivos lógicos internos - ILF (Internal Logical Files)

Es un grupo de datos relacionados de manera lógica o información de control significativos al usuario, residen dentro de la frontera de la aplicación y son mantenidos a través de procesos en el sistema que estamos midiendo.

Para que un archivo se considere lógico interno debe cumplir con las siguientes reglas:

- Ser un grupo de datos lógico o información de control significativos al usuario y que cumplen requerimientos específicos del mismo.
- El grupo de datos se mantiene dentro de la frontera de la aplicación que se está midiendo.
- El grupo de datos es mantenido a través de un proceso elemental de la aplicación.
- El grupo de datos identificado no ha sido contado como un archivo de interfaz externa (EIF) en la aplicación que se está midiendo.

El ejemplo más común es el de las tablas dentro de una base de datos relacional a las que nuestra aplicación hace referencia, archivos en un disco flexible, archivos planos en la base de datos de una PC, etc. Lo más importante, es que los archivos pueden mantenerse desde la aplicación que estamos midiendo, es decir, podemos agregar, cambiar y eliminar datos de esos archivos.

Archivos de interfaz externa - EIF (External Logical Files)

Es un grupo de datos relacionados de manera lógica, significativo al usuario, utilizado solamente para propósitos de referencia. Los datos residen completamente fuera de la aplicación que estamos midiendo y son mantenidos dentro de la frontera de otra aplicación. Un archivo de interfaz externa (EIF) contado en una aplicación debe ser un archivo lógico interno (ILF) en otra aplicación.

Para que un archivo se considere de interfaz externa se deben cumplir las siguientes reglas:

- Ser un grupo de datos lógico o información de control significativos al usuario y que cumplen requerimientos específicos del mismo.
- El grupo de datos es referenciado por, y externo a, la aplicación que se está midiendo.
- El grupo de datos no es mantenido por la aplicación que se está midiendo.
- El grupo de datos se cuenta como un ILF por al menos otra aplicación.
- El grupo de datos identificado no ha sido contado como ILF por la aplicación que se está midiendo.

Por ejemplo, en un sistema de clientes en una agencia de viajes, un ILF podría ser la tabla que contiene la información de los itinerarios dados a los clientes propios de la agencia. Un EIF podría ser la información con los itinerarios de los vuelos, utilizados para uso de la agencia, pero que seguramente son mantenidos por las líneas aéreas.

2.2.4 Funcionalidad de transacciones

Las transacciones representan la funcionalidad dada al usuario para procesar datos en una aplicación. Se definen como: entradas externas (EI), consultas externas (EQ) y salidas externas (EO).

Entradas externas - EI (External Inputs)

Una entrada externa es un proceso elemental donde un grupo de datos cruzan la frontera de la aplicación desde afuera hacia adentro. Los datos procesados mantienen uno o más archivos lógicos internos (ILF). Estos datos pueden obtenerse del usuario a partir de una pantalla de captura o desde otra aplicación.

Para considerar una entrada externa se deben cumplir las siguientes reglas:

- Los datos se reciben desde fuera de la frontera.
- Los datos en un ILF son mantenidos a través de un proceso elemental de la aplicación.
- Este proceso elemental es la unidad más pequeña de actividad que es significativa desde el punto de vista del usuario.
- El proceso elemental es autocontenido y deja autoconsistente la aplicación que se está midiendo.
- Los procesos identificados tienen que cumplir una de las siguientes reglas:
  - El proceso lógico es único dentro de la aplicación que se está midiendo.
  - Los datos identificados son diferentes de otras entradas dentro de la aplicación.

Ejemplos: Las pantallas de captura para mantener datos en un cierto archivo son el ejemplo clásico de un EI. También cuentan como entradas externas datos físicos que inician un proceso, como en el caso de un impulso de reloj (información de control), que en determinado instante hace que se active un proceso. Las entradas basadas en el manejo del ratón, entradas tomadas desde un disco flexible, etc.

Consultas externas - EQ (External Inquiries)

Una consulta externa es un proceso elemental compuesto de una combinación de una entrada externa y una salida externa que da como resultado una recuperación de datos a partir de uno o más archivos lógicos internos. El proceso de entrada no actualiza ningún archivo lógico interno, el proceso de salida no contiene datos calculados y no mantiene algún archivo lógico interno (ILF) de otra aplicación.

Para que se considere una consulta externa (EQ) se deben cumplir las siguientes reglas:

- La solicitud de entrada se introduce en la frontera de la aplicación.

- Los resultados de salida surgen desde dentro de la frontera de la aplicación.
- Los datos son recuperados.
- No contienen datos calculados.
- La solicitud de entrada y los resultados de salida muestran juntos un proceso elemental que es la unidad más pequeña de actividad significativa desde el punto de vista del usuario.
- Este proceso elemental es autocontenido y deja autoconsistente la aplicación que se está midiendo.
- El proceso elemental no actualiza un archivo lógico interno.
- Los procesos identificados tienen que cumplir una de las siguientes reglas:
  - El proceso lógico en el lado de entrada o de salida es único con respecto a otras consultas externas de la aplicación que se está midiendo.
  - Los elementos de datos identificados del lado de entrada o de salida son diferentes con respecto a otras consultas externas dentro de la aplicación.

Ejemplos: Funciones de usuario como vistas en una base de datos, búsquedas, despliegue, examinador (*browse*), mensajes de ayuda; una pantalla de inicio, etc. Un ejemplo muy común es abrir una pantalla de captura con datos ya desplegados y listos para cualquier cambio o simplemente como muestra.

Salidas externas - EO (External Outputs)

Una salida externa es un proceso elemental que genera datos o información de control enviados fuera de la frontera de la aplicación. Este proceso contiene datos calculados que se despliegan en pantalla, crean reportes o generan archivos de salida hacia otras aplicaciones. Estos reportes o archivos son creados a partir de uno o más archivos lógicos internos.

Para que un conjunto de datos se considere una salida externa (EO) se deben de cumplir las siguientes reglas:

- Los datos o información de control se envían fuera de la frontera.
- Los datos o información de control se envían a través de un proceso elemental de la aplicación.

- Este proceso elemental es la unidad más pequeña de actividad que es significativa al usuario.
- Este proceso elemental es autocontenido y deja un estado consistente la aplicación que se está midiendo.
- Los procesos identificados tienen que cumplir una de las siguientes reglas:
  - La lógica del proceso es única con respecto a otras salidas externas de la aplicación.
  - Los datos identificados son diferentes de otras salidas externas de la aplicación.

Ejemplos: Los reportes con datos procesados son el ejemplo más común. También pueden ser mensajes de información, datos que de alguna manera fueron calculados en la aplicación y se despliegan en una pantalla, datos que son transferidos a otras aplicaciones, etc. Lo importante es que sean tomados o calculados dentro de la aplicación que está siendo contada y se den a conocer fuera de la aplicación. La figura 2.1 muestra de una manera esquemática los conceptos anteriores.

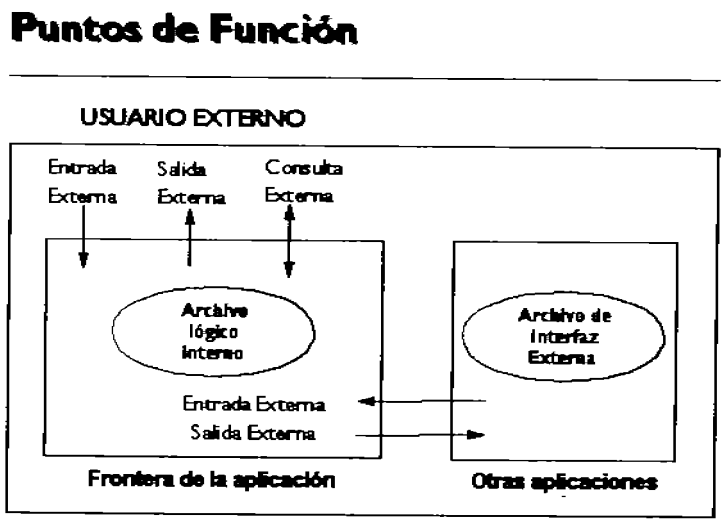


Figura 2.1 Componentes para calcular los puntos de función

2.2.5 Elementos para medir la complejidad

A los componentes anteriores se les asigna un grado de complejidad funcional basado en el número de los siguientes elementos: elementos de datos (DET), necesario para calcular la complejidad tanto en la funcionalidad de datos como en las transacciones; elementos de registro (RET), necesario para calcular la complejidad en las funcionalidades de datos y los archivos referenciados (FTR), necesarios para calcular la complejidad de las transacciones.

2.2.5.1 Elementos de dato - DET (Data Element Type)

La definición de los elementos de datos depende del contexto en el que sean considerados, sin embargo, en general se tratan de campos no recursivos, significativos al usuario.

DET en la funcionalidad de datos

Un elemento de datos (DET) en la funcionalidad de datos es un campo único no recursivo y significativo al usuario en el archivo lógico interno (ILF) o de interfaz externa (EIF).

Para contar un DET en la funcionalidad de datos debe de seguir las siguientes reglas:

- Contar un DET por cada campo significativo para el usuario y que no se repita en el ILF o EIF.
- Contar un DET por cada parte de un ILF o EIF que existe por que el usuario requiere una relación con otro ILF para ser mantenido. Ejemplo, llaves foráneas
- Contar como un solo DET los datos que aparecen más de una vez en un ILF o EIF por implementación técnica y los campos que son idénticos en forma y existen para permitir múltiples ocurrencias de los datos.

DET en una entrada externa

En el caso de una entrada externa es un campo mantenido en un archivo lógico interno (ILF) mediante el proceso de entrada; ya sea introducido por el usuario o a través de otra aplicación. Hay que contar como un elemento de dato (DET) la capacidad para especificar la acción a ser tomada por la entrada externa, por ejemplo un botón de comando.

Los elementos de datos en una entrada externa tienen que cumplir las siguientes reglas:

- Contar un DET por cada campo no recursivo, significativo al usuario y mantenido en un ILF por una entrada externa.
- Contar un DET por cada campo que aunque no sea introducido por el usuario mantiene a un ILF a través de una entrada externa. Ej. número de empleado generado por el sistema.
- Contar como un solo DET los campos lógicos que se almacenan físicamente como múltiples campos, pero para el usuario son una sola información.

DET en una salida externa

En el caso de una salida externa, un elemento de dato (DET) es un campo que aparece en la salida externa, ya sea proporcionado desde un archivo lógico interno (ILF) o calculado en la aplicación.

Para que un DET sea considerado en una salida externa debe cumplir las siguientes reglas:

- Contar un DET por cada campo no recursivo significativo al usuario que aparece en la salida externa.
- No contar las etiquetas y valores constantes como DET.
- No contar las variables de paginación o elementos generados por el sistema como DET. Ej. Fecha u hora generados por el sistema.
- Contar como un DET campos lógicos que físicamente se almacenan en varios campos.

2.2.5.2 Elementos de registro - RET (Record Element Type)

Son un subgrupo de campos significativo al usuario en un archivo lógico interno (ILF) o un archivo de interfaz externa (EIF). Existen dos tipos de subgrupos: opcional y obligatorio.

Los *opcionales* son los que el usuario tiene la opción de utilizar durante un proceso elemental que agrega o crea una instancia de los datos.

Los *obligatorios* son los que el usuario debe utilizar al menos una vez durante un proceso elemental que agrega o crea una instancia de los datos.

El conteo de los elementos de registro (RET) debe cumplir las siguientes reglas:

- Contar un RET por cada subgrupo opcional o mandatorio de un ILF o EIF.

- Si no existen subgrupos contar el ILF o EIF como un RET

Ejemplos: El tipo de elemento de registro es un subconjunto de tipos de elementos de datos. Por ejemplo, suponga que se desea almacenar información contenida en discos compactos (CD). Un disco compacto contiene la siguiente información: cantante, grupo, productor, nombre del disco, fecha y canciones. Por supuesto, existen muchas canciones en cada CD. Por cada canción, se tiene la siguiente información: nombre de la canción, autor y duración. En este caso el archivo que almacena esta información contiene 2 RET (la información del CD y la información de la canción) y 8 DET (5 DET del CD y 3 DET de las canciones).

2.2.5.3 Archivos referenciados - FTR (File Type Referenced)

Es un archivo lógico interno (ILF) o un archivo de interfaz externo (EIF) mantenido o leído por una transacción. Las reglas para determinar los archivos referenciados (FTR) dependen de la transacción a la que se hace referencia.

FTR en una entradas externa

En el caso de entradas externas (EI) las reglas son las siguientes:

- Contar un archivo referenciado por cada archivo lógico interno (ILF) mantenido en el proceso de entrada externa.
- Contar un archivo referenciado por cada archivo lógico interno (ILF) o de interfaz externa (EIF) leído durante el proceso de entrada externa.
- Contar un archivo referenciado por cada archivo lógico interno (ILF) que es mantenido y leído durante el proceso de entrada externa.

FTR en una consulta externa

En el caso de las consultas externas (EQ) se aplican las siguientes reglas:

- Contar un archivo referenciado por cada archivo lógico interno (ILF) o de interfaz externa (EIF) leído durante el proceso de salida externa ya sea del lado de entrada como del lado de salida.

FTR en una salida externa

En el caso de salidas externas (EO) se aplican las siguientes reglas:

- Contar un archivo referenciado por cada archivo lógico interno (ILF) o de interfaz externa (EIF) leído durante el proceso de salida externa.

Por ejemplo, en un sistema de recursos humanos, un archivo referenciado es la tabla o tablas en la base de datos que almacene todo lo referente a un empleado (contado ya como archivo lógico interno)

2.2.6 Las 14 características generales del sistema - GSC (General System Characteristics)

Para poder medir el sistema como un todo es necesario conocer las siguientes 14 características del sistema y el propósito de cada una de ellas.

1.- Comunicación entre datos

Saber que tanto el sistema requiere recursos de comunicación que permitan el intercambio de información, saber si las terminales conectadas de manera local a una unidad de control utilizan algún recurso de comunicación, o por ejemplo, saber si todos los vínculos de comunicación de datos requieren algún tipo de protocolo.

Por ejemplo, un programa de un banco multinacional que debe manejar transferencias monetarias electrónicas en instituciones financieras de todo el mundo, debe tener recursos de comunicación muy sofisticados.

2.- Procesamiento distribuido de datos

Saber si los datos distribuidos y las funciones en proceso son una característica de la aplicación dentro de la frontera.

Por ejemplo, un buscador web donde el proceso es realizado por más de una docena de servidores que trabajan en conjunto.

3.- Objetivos de desempeño

Conocer si el usuario estableció objetivos de desempeño precisos de la aplicación que tienen influencia en el diseño, desarrollo, instalación y ayuda de la aplicación.

Por ejemplo, un sistema de control de tráfico aéreo donde se requiere proporcionar de manera continua con precisión, la ubicación de una aeronave, desde un radar.

4.- Grado de configurabilidad

Saber si dentro de las consideraciones de diseño está el dar a la aplicación características de configuración adecuada para un sistema que va a tener un uso intenso.

Por ejemplo, un sistema en una universidad donde miles de estudiantes registran sus cursos de manera simultánea debe ser configurado con el propósito de tener un uso intenso en horas pico.

5.- Volumen de transacciones

Saber si existen muchas transacciones y por lo tanto se consideran en el diseño, desarrollo, instalación y soporte de la aplicación.

Por ejemplo, un programa bancario que realice millones de transacciones durante la noche para hacer un balance en todos los libros antes del siguiente día laborable.

6.- Captura de datos en-línea (interactiva)

Conocer si la aplicación captura datos y realiza funciones de control en línea.

Por ejemplo, un programa de solicitud de un préstamo hipotecario, en donde los interesados incorporan datos en un sistema informativo para llenar las formas de requisitos y ser posibles propietarios.

7.- Eficiencia para el usuario final

Ver si la aplicación está diseñada con el propósito de facilitarle al usuario final su manejo.

Por ejemplo, menús, manejo del ratón, pantallas de ayuda en línea, etc.

8.- Actualización de datos en línea

Saber si la aplicación cuenta con actualización en línea para los archivos lógicos internos.

Por ejemplo, el sistema de una aerolínea donde los agentes de viajes pueden reservar vuelos y asignar asientos. El software debe ser capaz de asegurar y modificar ciertos registros en la base de datos que aseguren el mismo asiento no se venderá dos veces.

9.- Complejidad del proceso

Determinar si una de las características de la aplicación es contar con un proceso complejo.

Por ejemplo, tiene un proceso matemático y lógico muy extenso que requirió de muchas horas de análisis.

10.- Reusabilidad

Medir que tanto la aplicación y su código se ha diseñado de manera específica para ser utilizado en otras aplicaciones.

Por ejemplo, un procesador de palabras diseñado para que su barra de menú pueda incorporarse en otras aplicaciones, como una hoja de cálculo o un generador de reportes.

11.- Facilidad de instalación y conversión

Conocer si la aplicación posee características para una fácil conversión e instalación. Saber si el plan y/o herramientas de conversión e instalación fueron proporcionadas y probadas durante la fase de prueba del sistema.

Por ejemplo, una aplicación de control de equipo que será instalado por personas no especializadas en un base petrolera mar dentro.

12.- Facilidad de operación

Saber si la aplicación puede operarse con facilidad. Si tiene procedimientos automatizados para ser instalado, poder hacer respaldos y recuperar datos.

Por ejemplo, en un programa donde se tiene que hacer una búsqueda histórica muy grande y procesa la información de tal manera que se minimice el número de veces que un operador cargue y descargue cintas (u otros dispositivos donde se almacena información) que contengan los datos.

13.- Posibilidad de instalación múltiple

Saber si la aplicación fue diseñada, desarrollada y tiene el soporte específico para ser instalada en múltiples sitios para muchas organizaciones.

Por ejemplo, una empresa multinacional en cuyo sistema de pago a la nómina considere las características distintas de varios países, incluyendo diversas modalidades y reglas de impuesto sobre la renta.

14.- Facilidad de modificación

Conocer si la aplicación fue diseñada, desarrollada y tiene el soporte específico para facilitar su cambio. Además de contar con la documentación necesaria para darle un mantenimiento seguro.

Por ejemplo, que exista un programa de predicción financiera que pueda adaptarse a las necesidades del administrador de un negocio particular, en la que necesitaría la información analizada por regiones geográficas y líneas de producto distintas a las especificadas en el programa original.

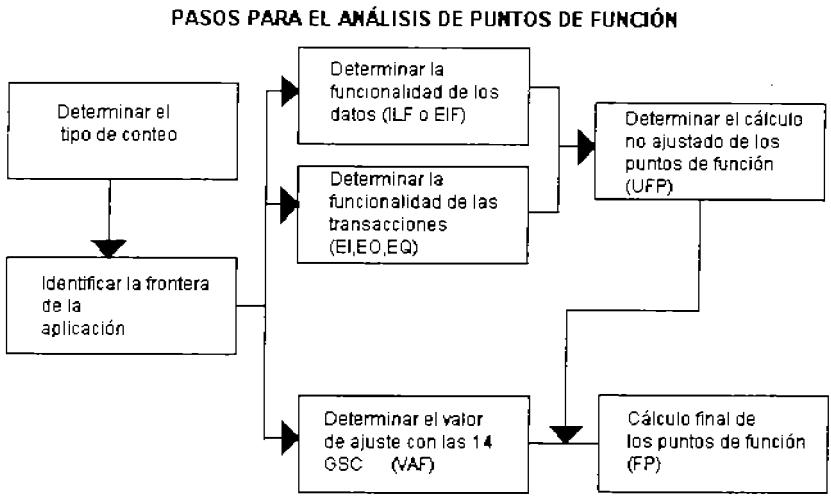
Dados estos conceptos, el siguiente capítulo muestra los pasos necesarios para calcular los puntos de función.

CAPÍTULO III
LA METODOLOGÍA PARA EL ANÁLISIS DE PUNTOS DE FUNCIÓN

Dadas las definiciones de los componentes necesarios para el cálculo, este capítulo explica los pasos para calcular los puntos de función. El capítulo describe con detalle como llevar el conteo de un proyecto que se lleva a cabo por primera vez, que es la base para poder hacer cualquier otro tipo de conteo. Posteriormente se menciona como es que se lleva a cabo el conteo tanto por mantenimiento como para actualizar el tamaño del proyecto.

3.1 PASOS EN EL ANÁLISIS DE PUNTOS DE FUNCIÓN

La siguiente figura muestra los pasos a seguir para determinar los puntos de función en una aplicación.



Paso 1. Determinar el tipo de conteo de puntos de función.

- Conteo de los proyectos por primera vez. En cuyo caso se hace el cálculo de toda la funcionalidad del sistema.

- Conteo por mantenimiento de los proyectos. En cuyo caso solamente se cuenta las funcionalidades agregadas, cambiadas o eliminadas del sistema.
- Conteo actualizado del proyecto. Proporciona el tamaño del proyecto ya con los cambios realizados.

Los cálculos preliminares de puntos de función son estimados de la funcionalidad que se entregará al cliente. Conforme se avanza en el proyecto y se desarrollan las funciones, es normal identificar funcionalidad adicional que no estaba en los requisitos originales. Este fenómeno se conoce como arrastre del alcance (*Scope Creep*). Es esencial actualizar el cálculo de puntos de función de la aplicación al término del proyecto. Si la aplicación cambia durante el desarrollo, el cálculo al final del ciclo de vida debe reflejar exactamente la funcionalidad entregada al usuario.

Paso 2. Identificar la frontera de la aplicación.

En base al límite entre la aplicación que está siendo medida y las aplicaciones externas a las que hace referencia y/o el dominio del usuario.

Paso 3. Determinar la funcionalidad de datos

a. Archivos lógicos internos (ILF)

b. Archivos de interfaz externa (EIF)

El número de ILFs y EIFs y su complejidad funcional relativa determinan la aportación de la funcionalidad de datos al cálculo de puntos de función sin ajustar. A cada uno de los ILF o EIF se les debe asignar una complejidad funcional basada en el número de elementos de datos (DET) y los elementos de registro (RET) asociados a este, mediante la tabla 3.1

Tabla 3.1 Matriz de complejidad de los archivos lógicos internos y de interfaz externa.

Número de elementos de Registro (RETs)	Número de elementos de datos (DETs)		
	1 - 19	20 - 50	'51 +
< 2	Baja	Baja	Promedio
2 - 5	Baja	Promedio	Alta
> 5	Promedio	Alta	Alta

Paso 4. Determinar la funcionalidad de transacciones

a. Entradas externas (EI)

El número de entradas externas (EI) y sus complejidades funcionales relativas determinan la aportación de ésta transacción al cálculo de puntos de función sin ajustar. A cada EI se le debe asignar una complejidad funcional basada en el número archivos referenciados (FTR) y los elementos de datos (DET) asociados a este, mediante la tabla 3.2

Tabla 3.2 Matriz de complejidad de la entradas externas

Número de archivos referenciados (FTRs)	Número de elementos de datos (DETs)		
	1 - 4	5 - 15	'16 +
< 2	Baja	Baja	Promedio
2	Baja	Promedio	Alta
> 2	Promedio	Alta	Alta

b. Salidas externas (EO)

El número de salidas externas (EO) y sus complejidades funcionales relativas determinan la aportación ésta transacción al cálculo de puntos de función sin ajustar. A cada EO se le debe asignar una complejidad funcional basada en el número archivos referenciados (FTR) y los elementos de datos (DET) asociados a este, mediante la tabla 3.3.

Tabla 3.3 Matriz de complejidad de la salidas externas

Número de archivos referenciados (FTRs)	Número de elementos de datos (DETs)		
	1 - 5	6 - 19	'20 +
< 2	Baja	Baja	Promedio
2 ó 3	Baja	Promedio	Alta
> 3	Promedio	Alta	Alta

c. Consultas externas (EQ)

El número de consultas externas y sus complejidades funcionales relativas determinan la aportación ésta transacción al cálculo de puntos de función sin ajustar. A cada EQ se le debe asignar una complejidad funcional dependiendo de sus entradas y salidas. Se calculan sus complejidades por separado, siguiendo las tablas 3.2 y 3.3 y se elige la que tenga más alta complejidad.

Paso 5. Determinar los puntos de función sin ajustar (UFP)

Por último, según el componente identificado junto con su complejidad, se le aplica el factor de ponderación indicado en la tabla 3.4. Y se suman para formar el valor de los puntos de función sin ajustar (UFP).

Tabla 3.4 Factor de ponderación de los componentes según su complejidad

Funcionalidad	Bajo	Promedio	Alto	Total
Archivos lógicos internos	___x 7	___x 10	___x 15	= _____
Archivos de interfaz externa	___x 5	___x 7	___x 10	= _____
Entradas externas	___x 3	___x 4	___x 6	= _____
Consultas externas	___x 3	___x 4	___x 6	= _____
Salidas externas	___x 4	___x 5	___x 7	= _____

Suma de puntos de función sin ajustar (UFP) =_____

Paso 6. Determinar el valor del factor de ajuste - 14 características generales del sistema.

Para calcular el factor de ajuste es necesario evaluar características generales del sistema como un todo. Estas características comprenden 14 factores que serán evaluados en términos de su grado de influencia en una escala de cero a cinco:

- **0 -** No presente, sin influencia alguna
- **1 -** Influencia incidental
- **2 -** Influencia moderada
- **3 -** Influencia promedio
- **4 -** Influencia significativa
- **5 -** Fuerte influencia

Esta evaluación puede modificar los puntos de función no ajustados dentro del rango ±35%. A continuación se dá una guía de cada una de las 14 características generales del sistema y en base a que asignar los valores. En caso de que ninguna de las descripciones se adecue con exactitud a la aplicación, se recurre al criterio para obtener el grado de influencia que mas se aproxime a la aplicación.

1.- Comunicación entre datos

¿Cuánta facilidad de comunicación existe para poder transferir o intercambiar información con la aplicación o el sistema?

Evaluar como:

0 - La aplicación es un proceso por lotes o reside solamente en una PC.

1 - La aplicación es por un proceso por lotes pero con entrada de datos o impresión remota.

2 - La aplicación es un proceso por lotes pero tiene entrada de datos e impresión remota.

3 - Conjunto de datos en línea o la presentación (*front-end*) es un protocolo de comunicación hacia un proceso por lotes o a un sistema de consulta.

4 - Más de un *front-end*, pero la aplicación soporta sólo un tipo de protocolo de comunicación de datos.

5 - Más de un *front-end* y la aplicación soporta más de un tipo de protocolo de comunicación de datos.

2.- Procesamiento distribuido de datos

¿Cómo se maneja la distribución de datos y el procesamiento de funciones?

Evaluar como:

0 - La aplicación no contempla la transferencia de datos o procesamiento de funciones entre los componentes del sistema.

1 - La aplicación prepara datos para que el usuario final los pueda procesar en otro componente del sistema como por ejemplo, una hoja de cálculo o un sistema de base de datos en una PC

2 - La aplicación prepara datos para transferir y procesar en otro componente del sistema (no para que un usuario final los pueda procesar)

3 - Los procesos distribuidos y la transferencia de datos son en línea y en una sola dirección.

4 - Los procesos distribuidos y la transferencia de datos son en línea y en ambas direcciones.

5 - Las funciones en proceso son ejecutadas en forma dinámica en los componentes del sistema.

3.- Objetivos de desempeño

¿Es el tiempo de respuesta requerido por el usuario?

Evaluar como:

0 - El usuario no establece algún requerimiento especial de desempeño.

1 - Se establecen y revisan requerimientos de desempeño y diseño, pero no se necesitan acciones especiales.

2 - El tiempo de respuesta es crítico en horas pico. No se requirió algún diseño especial para el uso del CPU.

3 - El tiempo de respuesta es crítico durante las horas laborables. No se requirió algún diseño especial para el uso del CPU.

4 - Los requerimientos de desempeño establecidos por el usuario son tan estrictos que requieren de tareas de análisis de desempeño en la fase de diseño.

5 - Además, se utilizaron herramientas de análisis de desempeño para las fases de diseño, desarrollo, y/o implementación para que coincidan con los requerimientos de desempeño establecidas por el usuario.

4.- Grado de configurabilidad

¿Existe un diseño de configuración especial para que el sistema tenga un uso intenso?

Evaluar como:

0 - No existen restricciones operativas explícitas o implícitas.

1 - Existen restricciones operativas, pero menos que en una aplicación típica, por lo que no se requiere un esfuerzo especial para cumplirlas.

2 - Existen algunas consideraciones de seguridad y tiempo.

3 - Existen requerimientos específicos de proceso para una parte de la aplicación.

4 - Se establecen restricciones operativas en la aplicación que hay que tomar en cuenta ya sea en el procesador central o en un procesador especial.

5 - Además, existen restricciones especiales en la aplicación en componentes distribuidos del sistema.

5.- Volumen de transacciones

¿ Con qué frecuencia se ejecutan las transacciones, diariamente, semanalmente, mensualmente, etc.

Evaluar como:

0 - No se prevé ningún periodo de transacción pico.

1 - Se prevé un periodo de transacción pico (por ejemplo, mensual, trimestral, por temporada o anual).

2 - Se prevé un periodo de transacción pico semanal.

3 - Se prevé un periodo de transacción pico diario.

4 - El usuario establece volúmenes de transacción pesadas en los requerimientos.

5 - Además, existen restricciones especiales en la aplicación en componentes distribuidos del sistema.

6.- Captura de datos en línea (interactiva)

¿Qué porcentaje de la información se captura en línea?

Evaluar como:

0 - Todas las transacciones son procesadas por lotes (batch mode).

1 - Del 1% al 7% de las transacciones se realizan mediante captura de datos interactiva.

2 - Del 8% al 15% de las transacciones se realizan mediante captura de datos interactiva.

3 - Del 16% al 23% de las transacciones se realizan mediante captura de datos interactiva.

4 - Del 24% al 30% de las transacciones se realizan mediante captura de datos interactiva.

5 - Más del 30% de las transacciones se realizan mediante captura de datos interactiva.

7.- Eficiencia para el usuario final

¿Se diseñó la aplicación para ser eficiente al usuario final?

Características:

- Menús
- Soporte en navegación (saltos, teclas de función, menús que se generan dinámicamente)
- Ayuda y documentación en línea
- Movimiento del cursor en forma automática
- Uso de barras de desplazamiento (*scrolling*)
- Impresión remota (vía transacciones en línea)
- Teclas de función presasignadas
- Sometimiento de los procesos batch desde transacciones en línea.
- Selección por medio del cursor de los datos en pantalla.
- Uso común de video inverso, sobresaltado, subrayado con colores y otros indicadores.
- Documentación detallada al usuario de las transacciones en línea.
- Interfaz con el ratón
- Ventanas *pop-up*
- Las menos pantallas posibles que acompañen a una función del negocio.
- Soporte bilingüe (para dos lenguajes, contar cuatro puntos)
- Soporte multilingüe (para más de dos lenguajes, contar seis puntos)

Evaluar como:

0 - En la aplicación no existe ninguna de las características anteriores.

1 - En la aplicación existen tres de las características anteriores.

2 - En la aplicación existen cuatro o cinco de las características anteriores.

3 - En la aplicación existen seis o más de las características anteriores, pero no existen requerimientos del usuario relacionados a la eficiencia.

4 - En la aplicación existen seis o más de las características anteriores y ademas se establecieron requerimientos para la eficiencia del usuario lo suficientemente fuertes para necesitar tareas de diseño. Por ejemplo, minimizar el uso del teclado, maximizar opciones por omisión (*default*), uso de plantillas, etc.

5 - En la aplicación existen seis o más de las características anteriores y ademas se establecieron requerimientos para la eficiencia del usuario lo suficientemente fuertes para necesitar herramientas y procesos especiales a fin de demostrar que los objetivos han sido alcanzados.

8.- Actualización de datos en línea

¿Cuántos archivos lógicos internos son actualizados por transacciones en línea?
Evaluar como:

0 - Ninguna

1 - Existe actualización en línea en uno a tres archivos de control. El volumen de la actualización es bajo y la recuperación es sencilla.

2 - Existe actualización en línea en cuatro o más archivos de control. El volumen de la actualización es bajo y la recuperación es sencilla..

3 - Existe actualización en línea en la mayor parte de los archivos lógicos internos.

4 - Además, la protección contra pérdida de datos es esencial y es diseñado y programado especialmente en el sistema.

5 - Además, existen procedimientos automatizados de recuperación de datos con un mínimo de intervención del operador.

9.- Complejidad del proceso

¿ Tiene la aplicación un proceso lógico o matemático complicado?
Características:

- Control sensible (por ejemplo, proceso de auditoría especial) y/o proceso de seguridad específica.
- Proceso lógico extenso.

• Proceso matemático extenso.

• Existen varios procesos por excepciones que dan como resultado transacciones incompletas que deben ser nuevamente procesadas; por ejemplo, transacciones ATM incompletas ocasionadas por interrupciones de teleproceso, datos perdidos o ediciones incorrectas.

• Proceso complejo por manejar múltiples posibilidades entrada/salida; por ejemplo, multi-media, independencia de dispositivos.

Evaluar como:

0 - Ninguna de las características anteriores.
1 - Una de las características anteriores.
2 - Dos de las características anteriores.
3 - Tres de las características anteriores.
4 - Cuatro de las características anteriores.
5 - Las cinco anteriores.

10.- Reusabilidad

¿ Se desarrolló la aplicación para satisfacer necesidades de uno o muchos usuarios?
Evaluar como:

0 - No se utiliza ni se produce código reusable.
1 - Se utiliza código reusable dentro de la aplicación.
2 - Menos del 10% de la aplicación considera las necesidades de más de un usuario.
3 - El 10% o más de la aplicación considera las necesidades de más de un usuario.
4 - La aplicación es empacada y/o documentada de manera específica para ser fácilmente reusable y la aplicación es personalizada por el usuario a nivel de código.
5 - La aplicación es empacada y/o documentada de manera específica para ser fácilmente reusable y la aplicación es personalizada por medio de parámetros por el usuario.

11.- Facilidad de instalación y conversión

¿Qué tan difícil es la instalación y conversión del sistema?
Evaluar como:

0 - No existen consideraciones especiales establecidas por el usuario ni la aplicación requiere de ningún programa de instalación (*set-up*).

1 - No existen consideraciones especiales establecidas por el usuario pero la aplicación requiere de un programa de instalación.

2 - El usuario establece requerimientos de conversión e instalación. Se proporcionan y prueban guías de instalación. No se considera importante el impacto de la conversión en el proyecto.

3 - El usuario establece requerimientos de conversión e instalación. Se proporcionan y prueban guías de instalación. Se considera importante el impacto de la conversión en el proyecto.

4 - Además de (2), se proporcionan y prueban herramientas de conversión e instalación automáticas.

5 - Además de (3), se proporcionan y prueban herramientas de conversión e instalación automáticas. por el usuario.

12.- Facilidad de operación

¿Qué tan efectivo y/o automatizado son los procedimientos de arranque, recuperación y respaldo?

Evaluar como:

0 - El usuario no establece consideraciones de operación especiales distintos a los procedimientos de respaldo normales.

1 - 4 Elija las siguientes características, que apliquen a su sistema. Cada una de las características cuenta un punto, a menos se indique lo contrario.

- Se proporcionan procesos efectivos de arranque, respaldo y recuperación pero se requiere de la intervención de un operador.
- Se proporcionan procesos efectivos de arranque, respaldo y recuperación pero no se requiere de la intervención de un operador. (Cuenta dos puntos.)
- La aplicación minimiza la necesidad del montaje de cintas.
- La aplicación minimiza la necesidad del manejo de papel.

5 - La aplicación está diseñada para operarse sin la intervención de un operador, excepto para iniciar y finalizar la aplicación. La aplicación recupera errores de manera automática.

13.- Posibilidad de instalación múltiple

¿La aplicación fue diseñada, desarrollada y tiene el soporte específico para ser instalada en múltiples sitios para muchas organizaciones?

Evaluar como:

0 - No existen requerimientos del usuario que consideren la necesidad de instalar en más de un sitio.

1 - Se considera en el diseño la necesidad de que la aplicación se ejecute en múltiples sitios y la aplicación se diseña para operar sólo bajo ambientes idénticos de software y hardware.

2 - Se considera en el diseño la necesidad de que la aplicación se ejecute en múltiples sitios y la aplicación se diseña para operar sólo bajo ambientes similares de software y/o hardware.

3 - Se considera en el diseño la necesidad de que la aplicación se ejecute en múltiples sitios y la aplicación se diseña para operar bajo ambientes diferentes de software y hardware.

4 - Se proporciona y prueba un plan de documentación y soporte para que una aplicación como la descrita en (1), se pueda ejecutar en múltiples sitios.

5 - Se proporciona y prueba un plan de documentación y soporte para que una aplicación como la descrita en (2), se pueda ejecutar en múltiples sitios.

14.- Facilidad de modificación

¿La aplicación fue diseñada, desarrollada y tiene el soporte específico para facilitar su cambio?

Evaluar como:

0 - No existen requerimientos del usuario que consideren la necesidad de diseñar la aplicación para minimizar o facilitar un cambio.

1 - 5 Elija las siguientes características, que apliquen a su sistema.

- Tiene la facilidad de consultar/reportar de manera flexible por lo que maneja solicitudes simples; por ejemplo un y/o lógico aplicados a un sólo archivo lógico interno (cuenta un punto).
- Tiene la facilidad de consultar/reportar de manera flexible por lo que maneja solicitudes de complejidad promedio; por ejemplo un y/o lógico aplicados a más de un archivo lógico interno (cuenta dos punto).
- Tiene la facilidad de consultar/reportar de manera flexible por lo que maneja solicitudes complejas; por ejemplo combinaciones de y/o lógico en uno o más archivos lógicos internos (cuenta tres punto).
- El control de datos se guardan en tablas mantenidas por el usuario con procesos interactivos en línea pero los cambios tienen efecto sólo al siguiente día laborable.
- El control de datos se guardan en tablas mantenidas por el usuario con procesos interactivos en línea pero los cambios tienen efecto inmediatamente (cuenta dos puntos).

La suma de los niveles de influencia de las 14 características generales dan como resultado el **grado de influencia total - TDI** (***Total Degree Influence***).

El **valor del factor de ajuste - VAF** (***Value Adjustment Factor***) es obtenida mediante la siguiente fórmula:

$$VAF = 0.65 + (0.01 \times TDI)$$

Paso 6. Calcular el total de puntos de función.

Aplicando el valor de ajuste, los puntos de función sufrirán una variación de ±35%. Esto es, suponiendo que existe influencia cero de las 14 características generales, el total de puntos de función se decrementará en un 35%, en caso de que exista un total de influencia de 70, el total de puntos de función se incrementará en un 35%. Por lo que el **total de puntos de función (FP)** de la aplicación que se cuenta por primera vez es:

FP = UFP x VAF

3. 2 CONTEO POR MANTENIMIENTO DEL PROYECTO

En el caso del conteo por mantenimiento de los proyectos, es necesario aplicar la siguiente fórmula.

EFP = [(ADD) + CHGA) x VAFA] + (DEL x VAFB)

Donde:

EFP (Enhancement Function Point) = Puntos de función por mantenimiento del proyecto.

ADD = Puntos de función sin ajustar de aquellas funciones que fueron **agregadas** al hacer el cambio en el proyecto.

CHGA = Puntos de función sin ajustar de aquellas funciones que fueron **modificadas** al hacer el cambio en el proyecto. Este número refleja las funciones **después** de las modificaciones.

VAFA = Factor de valor de ajuste de la aplicación **después** de realizar los cambios en el proyecto.

DEL = Puntos de función sin ajustar de aquellas funciones que fueron **eliminadas** al hacer el cambio en el proyecto.

VAFB = Factor de valor de ajuste de la aplicación **antes** de realizar los cambios en el proyecto.

3.3 CONTEO ACTUALIZADO DEL PROYECTO

Al hacer modificaciones en el proyecto, la funcionalidad de la aplicación cambia y es por eso que el conteo debe ser actualizado. Es decir:

- Al agregar (nueva) funcionalidad el tamaño de la aplicación se incrementa.
- Al cambiar funcionalidad el tamaño de la aplicación podría incrementarse, decrementarse o no tener efecto alguno.
- Al eliminar funcionalidad el tamaño de la aplicación se decrementa.

La fórmula para calcular el total de puntos de función de la aplicación con los cambios realizados es la siguiente:

FP = [(UFPB + ADD + CHGA) - (CHGB + DEL)] x VAFA

Donde:

FP = Total de puntos de función

UFPB = Puntos de función sin ajustar contados antes de los cambios en el proyecto.

ADD = Puntos de función sin ajustar de aquellas funciones que fueron **agregadas** al hacer el cambio en el proyecto.

CHGA = Puntos de función sin ajustar de aquellas funciones que fueron **modificadas** al hacer el cambio en el proyecto. Este número refleja las funciones **después** de las modificaciones.

CHGB = Puntos de función sin ajustar de aquellas funciones que fueron **modificadas** al hacer el cambio en el proyecto **antes** de que los cambios fueran realizados.

DEL = Puntos de función sin ajustar de aquellas funciones que fueron **eliminadas** al hacer el cambio en el proyecto.

VAFA = Factor de valor de ajuste de la aplicación **después** de realizar los cambios en el proyecto.

En el anexo se presentan dos hojas guías para el cálculo del análisis de puntos de función.

CAPÍTULO IV
EJEMPLO DEL ANÁLISIS DE PUNTOS DE FUNCIÓN

El siguiente es un ejemplo de conteo de puntos de función:

Un hospital desea automatizar su sistema de ingresos de pacientes, de los médicos que laboran en el hospital y de historias médicas. Este sistema tendrá acceso a dos sistemas ya existentes del mismo hospital y que sólo utilizará como referencia, que son: el sistema que lleva el control de las áreas o especialidades del hospital y el que almacena la información de los padecimientos más comunes. El ejemplo cuenta los puntos de función basándose en los requerimientos y su diagrama de entidad-relación.

1. Pacientes
    a. Ingresar un paciente (Añadirlo en la base de datos) introduciendo:
        - Datos particulares del paciente.
        - Padecimiento (validar con el sistema de padecimientos más comunes).
        - Médico asignado
    b. Cambiar datos de un paciente.
        - Actualizar cualquier información con respecto a un paciente excepto la clave del paciente.
    c. Eliminar un paciente.
        - Eliminar toda la información con respecto a un paciente.
    d. Solicitar los datos de un paciente.
        - Dada la clave del paciente, consultar la información de un paciente en particular. De cada paciente se desplegará además de sus datos particulares, el nombre de su médico asignado y su padecimiento
    e. Solicitar un reporte con todos los pacientes.
        - De cada paciente se desplegará además de sus datos particulares, el nombre de su médico asignado y su padecimiento. Se desplegará también, el total de todos los pacientes.
2. Médicos
    a. Añadir los datos de un médico con la siguiente información:
        - Datos particulares del médico
        - Área de especialidad (validar con el archivo que almacena las especialidades)

b. Cambiar los datos de un médico.

- Modificar los datos de un médico en particular excepto su clave de médico.

c. Eliminar un médico.

- Eliminar toda la información con respecto a un médico en particular.

d. Solicitar los datos de un médico.

- Dada la clave del médico, solicitar información detallada de un médico en particular. Debe incluir el nombre de la especialidad.

e. Solicitar el reporte de todos los médicos que laboran en el hospital. Este desplegará con el total de todos los médicos.

3. Historia médica

a. Añadir una historia médica con la siguiente información.

- Clave del paciente.
- Descripción.

b. Cambiar la información de una historia médica.

- Mediante la clave del paciente, desplegar su historia médica y poder cambiar la información, excepto la clave del paciente.

c. Eliminar una historia médica.

- Mediante la clave del paciente, eliminar su historia médica.

d. Consultar una historia médica.

- Mediante la clave del paciente, generar un reporte en pantalla que incluya nombre del paciente, nombre del médico, nombre del padecimiento y la historia médica.

4. Consultar y reportar información sobre las áreas de especialidad del hospital.

a. Consultar en pantalla la información de las áreas de especialidad.

b. Hacer un reporte impreso de las áreas de especialidad.

5. Otras características del sistema de ingreso al hospital serán las siguientes:

a. La aplicación tiene más de un *front-end* pero soporta sólo un tipo de protocolo de comunicación TP.

b. No será un proceso distribuido.

c. No existen requerimientos de desempeño especiales.

d. No existen restricciones operativas explícitas o implícitas.

e. No existe un período de transacción pico.

f. La mayoría de las transacciones se realizan mediante captura de datos interactiva.

g. Para eficiencia del usuario el sistema tendrá movimiento del cursor automatizado, uso de ratón y elección de pantalla por medio del cursor.

h. Actualización en línea todos los archivos de lógicos, pero no hay protección contra pérdida de datos.

i. No existen procesos complejos.

j. La mayoría (más del 50%) de la aplicación y código debe ser designado, desarrollado y soportado para utilizarse en otras aplicaciones.

k. Ningún requerimiento de conversión o instalación especial.

l. La aplicación minimizará la necesidad tanto del manejo de cintas y papel. Tiene un proceso de inicio, respaldo y recuperación de datos, pero se requiere de la intervención de un operador.

m. Se debe considerar en el diseño, la necesidad de instalar la aplicación en distintos lugares donde operará bajo ambientes de hardware y software similares.

n. Los datos de control serán mantenidos de manera interactiva y toman efecto de inmediato. No existe una capacidad flexible de consulta/reporte.

Atributos de cada entidad

PACIENTE	**DATOS PARTICULARES DEL PACIENTE**
ClavePaciente (K)	*ClavePaciente(K)*
Cuarto	*NombrePaciente*
FechaIngreso	*Edad*
ClaveMedico (FK)	*EstadoCivil*
ClavePadecimiento (FK)	*Direccion*
	Telefono
HISTORIA MEDICA	**MEDICO**
ClavePaciente (FK)	*ClaveMedico (K)*
Historia	*NombreMedico*
	ClaveEspecialidad (FK)
	Horario
PADECIMIENTOS	**ESPECIALIDADES**
ClavePadecimiento (K)	*ClaveEspecialidad (K)*
Padecimiento	*Especialidad*
DescripcionPadecimiento	
ClaveEspecialidad (FK)	

donde K = llave y FK = llave foránea

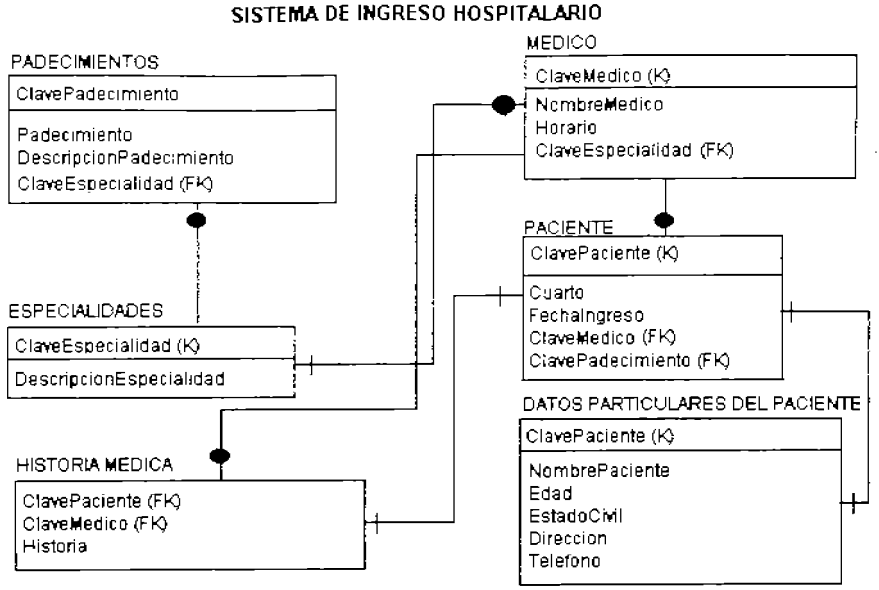


Figura 4.1. Diagrama de entidad-relación del sistema de ingreso hospitalario.

Con los requerimientos anteriores y el diagrama de entidad-relación de la figura 4.1 se construyó el siguiente modelo jerárquico de procesos:

INGRESO HOSPITALARIO

1. ADMINISTRAR_PACIENTE
   a. AGREGAR_PACIENTE
   b. ACTUALIZAR_PACIENTE
   c. CONSULTAR_PACIENTE
   d. ELIMINAR_PACIENTE
   e. REPORTAR_PACIENTES (contiene datos calculados)
2. MANTENER_MEDICOS
   a. AGREGAR_MEDICO
   b. ACTUALIZAR_MEDICO
   c. CONSULTAR_MEDICO

d. ELIMINAR_MEDICO
e. REPORTAR_MEDICOS (contiene datos calculados)

3. MANTENER_HISTORIAMEDICA
   a. AGREGAR_HISTORIAMEDICA
   b. ACTUALIZAR_HISTORIAMEDICA
   c. CONSULTAR_HISTORIAMEDICA
   d. ELIMINAR_HISTORIAMEDICA
4. CONSULTAR_ESPECIALIDADES
   a. CONSULTAR_ESPECIALIDADES
   b. REPORTAR_ESPECIALIDADES

Con la información anterior y siguiendo los pasos expuestos en el capítulo anterior es posible hacer el análisis de puntos de función:

Paso 1. Determinar el tipo de conteo.

Se supone que esta es la primera vez que se va a llevar a cabo el conteo. Por lo tanto el cálculo es de todo el proyecto.

Paso 2. Determinar la frontera de la aplicación.

Es necesario distinguir entre la aplicación que se está midiendo y las aplicaciones a las que solamente se hace referencia. Los requerimientos ayudan a delimitar esta frontera, ya que se hace explícito que las entidades PADECIMIENTOS y ESPECIALIDADES son sólo de referencia y no son mantenidos por el sistema que se está contando. Observando el diagrama de entidad-relación se separan las entidades que la aplicación va a mantener (Figura 4.2).

Las entidades que se encuentran dentro de la frontera de la aplicación a contar son: PACIENTE, DATOS PARTICULARES DEL PACIENTE, MEDICO, HISTORIA MEDICA. Las que quedan fuera de la frontera son PADECIMIENTOS y ESPECIALIDADES

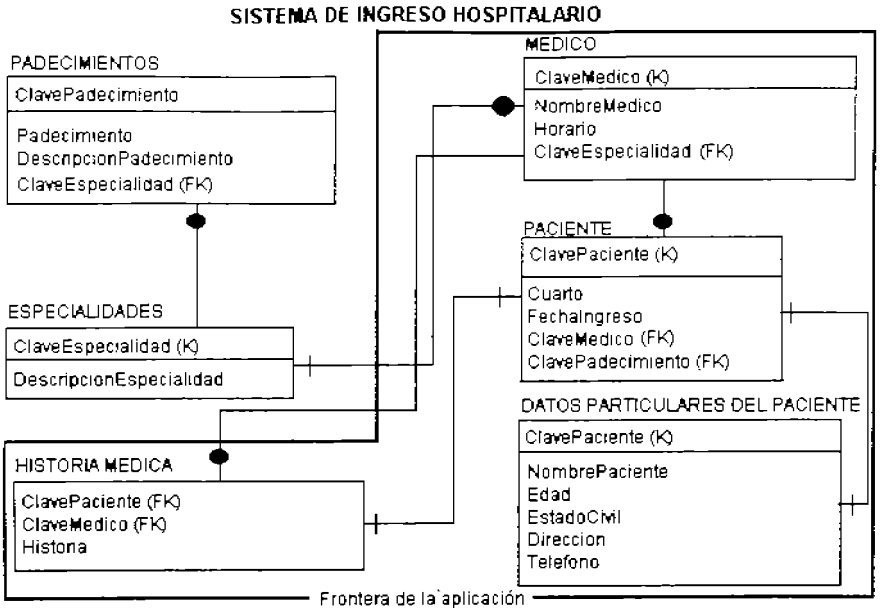


Figura 4.2. Frontera de la aplicación del sistema de ingreso hospitalario.

Paso 3. Identificar la funcionalidad de datos.

Con este paso se identifica lo que son los archivos lógicos internos (ILF) y los archivos de interfaz externa (EIF) junto con los componentes necesarios para evaluar sus complejidades: elementos de registro (RET) y elementos de datos (DET).

Archivos lógicos internos (ILF)

Los archivos lógicos internos se encuentran dentro de la frontera de la aplicación. Por lo que los candidatos son: PACIENTE, DATOS PARTICULARES DEL PACIENTE, MEDICOS e HISTORIA MEDICA.

Se puede utilizar una tabla como la siguiente con cada candidato para saber si cumple con las reglas para ser un archivo lógico interno. Por ejemplo, con las entidades PACIENTE e HISTORIA MEDICA.

Entidad	Reglas de conteo de ILF	¿Aplica?
PACIENTE	El grupo de datos o información de control es un grupo de datos lógico, significativo al usuario que satisface requerimientos específicos del usuario	No, ya que debe incluir DATOS PERSONALES DEL PACIENTE para poder representar el requerimiento del usuario y así poder agregar toda la información sobre el paciente.
	El grupo de datos es mantenido dentro de la frontera de la aplicación.	Sí
	El grupo de datos es mantenido o modificado a través de un proceso elemental de la aplicación.	No, no existe un proceso que mantiene solamente a la entidad PACIENTE. El usuario debe mantener en el proceso elemental a PACIENTE y a DATOS PERSONALES DEL PACIENTE.
	El grupo de datos identificado no ha sido contado como un EIF para la aplicación.	Sí, la regla aplica ya que los datos no han sido contados como un EIF para otra aplicación.

Entidad	Reglas de conteo de ILF	¿Aplica?
HISTORIA MEDICA	El grupo de datos o información de control es un grupo de datos lógico, significativo al usuario que satisface requerimientos específicos del usuario	Sí
	El grupo de datos es mantenido dentro de la frontera de la aplicación.	Sí
	El grupo de datos es mantenido o modificado a través de un proceso elemental de la aplicación.	Sí, el proceso es introducir la información de la historia médica de un paciente en el sistema de ingreso hospitalario.
	El grupo de datos identificado no ha sido contado como un EIF para la aplicación.	Sí, la regla aplica ya que los datos no han sido contados como un EIF para otra aplicación.

Si se juntan las entidades de PACIENTE con DATOS PARTICULARES DEL PACIENTE, entonces sí aplican las reglas para archivos lógicos internos. De esta forma se identifican tres ILF:

1) PACIENTE y DATOS PARTICULARES DEL PACIENTE que en adelante se le llama PACIENTE para generalizar.

2) MEDICO

3) HISTORIA MEDICA

Archivos de interfaz externa (EIF)

Las dos entidades que quedaron fuera de la frontera de la aplicación son los candidatos a ser archivos de interfaz externa. Estos son ESPECIALIDADES Y PADECIMIENTOS.

Se puede utilizar una tabla como la siguiente con cada candidato para saber si las entidades candidatas cumplen con las reglas para archivos de interfaz externa. Se dá el ejemplo con la entidad de ESPECIALIDADES.

Entidad	Reglas de conteo de EIF	¿Aplica?
ESPECIALIDADES	El grupo de datos o información de control es un grupo de datos lógico, significativo al usuario que satisface requerimientos específicos del usuario	Sí, los usuarios necesitan poder obtener información sobre las especialidades que existen en el hospital.
	El grupo de datos es referenciado por, y externo a, la aplicación que se está contando.	Sí, solamente se usa como referencia.
	El grupo de datos **no** es mantenido por la aplicación que está siendo contada	Sí, es identificado como un proceso elemental en otro sistema.
	El grupo de datos es contado como un ILF por al menos otra aplicación.	Sí, en el sistema que lo mantiene
	El grupo de datos identificado no ha sido contado como un ILF por la aplicación	Sí.

Si se aplican las reglas a la entidad PADECIMIENTOS, también se cumplen todas. Así, identificamos dos EIF:

1) ESPECIALIDADES

2) PADECIMIENTOS

Para medir la complejidad de cada una de las funcionalidades de datos, es necesario identificar dos elementos para cada una de ellas: 1) de registro (RET) y 2) de datos (DET).

Elementos de registro (RET)

ILF	Subgrupos
PACIENTE	PACIENTE y DATOS PARTICULARES DEL PACIENTE.
MEDICO	Ninguno, cuenta como un RET

HISTORIA MEDICA	Ninguno, cuenta como un RET

EIF	Subgrupos
PADECIMIENTOS	Ninguno, cuenta como un RET
ESPECIALIDADES	Ninguno, cuenta como un RET

La cuenta de RET para cada una de las funcionalidades de datos es la siguiente:

Entidad	Funcionalidad de datos	Número de RET
PACIENTE	ILF	2
MEDICO	ILF	1
HISTORA MEDICA	ILF	1
PADECIMIENTOS	EIF	1
ESPECIALIDADES	EIF	1

Elementos de datos (DET)

Para ver el número de DET en cada una de las funcionalidades identificadas, se recurre al diagrama de entidad-relación para determinar los campos con los que cuenta cada una de estas entidades. Una tabla como la siguiente puede ser de utilidad:

Entidades	¿Es un campo significativo al usuario en el ILF? Si, sí, contar como un DET.	¿Es una llave foránea? Si lo es, contar como un DET.	Contar la implementación física como un DET sencillo.
PACIENTE			
ClavePaciente	Sí	No	No
Cuarto	Sí	No	No
FechaIngreso	Sí	No	No
ClaveMedico	No	Sí	No
ClavePadecimiento	No	Sí	No
NombrePaciente	Sí	No	No
Edad	Sí	No	No
EstadoCivil	Sí	No	No
Direccion	Sí	No	No
Telefono	Sí	No	No
Total de DET	8	2	

En este caso como se unió DATOS PARTICULARES DEL PACIENTE a PACIENTE, se cuentan todos los campos de la información del PACIENTE como parte de un sólo ILF.

Si se analiza cada uno de los ILF e EIF identificados, da como resultado el siguiente número de DET de cada uno:

Entidad	Funcionalidad de datos	Número de DET
PACIENTE	ILF	10
MEDICO	ILF	4
HISTORIA MEDICA	ILF	3
PADECIMIENTOS	EIF	4
ESPECIALIDADES	EIF	2

Paso 4. Identificar la funcionalidad de transacciones.

Este paso determina la funcionalidad de transacciones: entradas externas (EI), salidas externas (EO) y consultas externas (EQ). Junto con los componentes necesarios para evaluar sus complejidades: Elemento de Datos (DET) y Archivos Referenciados (FTR). Para esto, es necesario observar los requerimientos del usuario y el modelo jerárquico de procesos.

Entradas Externas - EI

A partir del modelo jerárquico de procesos, se identifica los que mantendrán a alguno de los ILF identificados. Ya identificados, se aplica a cada uno de ellos las reglas de entradas externas para ver si efectivamente lo son ayudados de una tabla como la siguiente:

Modelo del proceso	Reglas de conteo de EI	¿Aplica?
1. ADMINISTRAR_PACIENTE a. AGREGAR_PACIENTE	Los datos son recibidos desde fuera de la frontera de la aplicación.	Sí
	Los datos de un ILF son mantenidos a través de un proceso elemental de la aplicación.	Sí, se mantiene al ILF PACIENTE (incluyendo DATOS PERSONALES DEL PACIENTE)
	El proceso es la unidad de actividad más pequeña significativa desde la perspectiva del usuario.	Sí, el usuario necesita crear, o agregar un paciente.
	El proceso se autocontiene y deja al negocio en un estado consistente.	Sí
	Para el proceso identificado, aplica una de las dos reglas siguientes:	
	- La lógica del proceso es único con respecto a otras entradas externas.	Sí
	- Los elementos de datos son diferentes con respecto a otras entradas externas.	Sí

Si se aplica a cada uno de los procesos las reglas de entradas externas, resultan los siguientes procesos:

1.a AGREGAR_PACIENTE
1.b ACTUALIZAR_PACIENTE
1.d ELIMINAR_PACIENTE
2.a AGREGAR_MEDICO
2.b ACTUALIZAR_MEDICO
2.d ELIMINAR_MEDICO
3.a AGREGAR_HISTORIAMEDICA
3.b ACTUALIZAR_HISTORIAMEDICA
3.d. ELIMINAR_HISTORIAMEDICA

Consultas Externas - EQ

Ahora se analiza cuál de los procesos que no fueron entradas externas pueden ser consultas externas. Y se aplican las reglas con una tabla como la siguiente:

Modelo del proceso	Reglas de conteo de EQ	¿Aplica?
1. ADMINISTRAR_PACIENTE c. CONSULTAR_PACIENTE	Una solicitud de entrada se introduce en la frontera de la aplicación.	Sí, por ejemplo un clic para solicitar la consulta
	El resultado sale de la frontera de la aplicación.	Sí
	Se recuperan datos.	Sí.
	Los datos recuperados no contienen dados derivados o calculados.	Sí
	El proceso es la unidad de actividad más pequeña que es significativa al usuario final del negocio.	Sí, el usuario necesita ver la información de un paciente.
	El proceso se autocontiene y deja al negocio en un estado consistente.	Sí
	El proceso no actualiza un ILF	Sí
	Para el proceso identificado, aplica una de las dos reglas siguientes:	
	- La lógica del proceso es única con respecto a otras consultas externas.	Sí
	- Los elementos de datos son diferentes con respecto a otras consultas externas.	Sí

Aplicando las reglas anteriores, se identifican estas consultas externas:

1.c CONSULTAR_PACIENTE

2.c CONSULTAR_MEDICO

3.c CONSULTAR_HISTORIAMEDICA

4.a CONSULTAR_ESPECIALIDADES

4.b REPORTAR_ESPECIALIDADES

El proceso de REPORTAR_ESPECIALIDADES es incluido como consulta externa porque no contiene datos calculados.

Salidas Externas - EO

Por último, hay que identificar si los procesos restantes son salidas externas. Para esto es necesario aplicar las reglas correspondientes con una tabla como la siguiente:

Modelo del proceso	Reglas de conteo de EO	¿Aplica?
1. ADMINISTRAR_PACIENTE e. REPORTAR_PACIENTES	El proceso envía datos o información de control externa a la frontera de la aplicación.	Sí
	El proceso es la unidad de actividad más pequeña que es significativa al usuario final del negocio..	Sí, el usuario necesita ver la información de los pacientes.
	El proceso se autocontiene y deja al negocio en un estado consistente.	Sí.
	Para el proceso identificado, aplica una de las dos reglas siguientes:	
	- La lógica del proceso es única con respecto a otras salidas externas.	Sí
	- Los elementos de datos son diferentes con respecto a otras salidas externas.	Sí

Aplicando las reglas a los procesos restantes, se identifican las siguientes salidas externas:

1.e REPORTAR_PACIENTES

2.e REPORTAR_MEDICOS

Es importante hacer notar que estos reportes contienen datos calculados, a diferencia de los reportes identificados como consultas externas.

Para medir la complejidad de cada una de las funcionalidades de transacciones, se identifican dos elementos para cada una de ellas: 1) archivos de referencia (FTR) y 2) de datos (DET).

Archivos referenciados (FTR)

Para identificar los archivos referenciados para cada uno de los procesos, es necesario referirse a las funcionalidades de datos y observar a cuáles mantiene o hace referencia cada uno de los procesos. Se pueden aplicar las reglas de FTR en cada uno de las transacciones con tablas como las siguientes:

FTR para entradas externas

Proceso	Contar cada ILF sólo mantenido	Contar cada ILF o EIF sólo referenciado	Contar sólo 1 FTR por ILF referenciado y mantenido
1.a AGREGAR_ PACIENTE	PACIENTE	PADECIMIENTOS, MEDICOS	
1.b ACTUALIZAR_ PACIENTE		PADECIMIENTOS, MEDICOS	PACIENTE
1.d ELIMINAR_ PACIENTE			PACIENTE
2.a AGREGAR_ ,MEDICO	MEDICO	ESPECIALIDADES	
2.b ACTUALIZAR_ MEDICO		ESPECIALIDADES	MEDICO
2.d ELIMINAR_ MEDICO			MEDICO
3.a AGREGAR_ HISTORIAMEDICA	HISTORIA MEDICA		
3.b ACTUALIZAR_ HISTORIAMEDICA			HISTORIA MEDICA
3.d ELIMINAR_ HISTORIAMEDICA			HISTORIA MEDICA

FTR para consultas externas

Consulta externa	Para la entrada contar cada ILF o EIF referenciado	Para la salida contar cada ILF o EIF referenciado
1.c CONSULTAR_PACIENTE	PACIENTE	PACIENTE, MEDICO PADECIMIENTOS
2.c CONSULTAR_MEDICO	MEDICO	MEDICO ESPECIALIDAD
3.c CONSULTAR_ HISTORIAMEDICA	PACIENTE	HISTORIA MEDICA PACIENTE MEDICO, PADECIMIENTO
4.a CONSULTAR_ ESPECIALIDADES	ESPECIALIDADES	ESPECIALIDADES
4.b REPORTAR_ ESPECIALIDADES	ESPECIALIDADES	ESPECIALIDADES

FTR para salidas externas

Salidas externas	Contar cada ILF o EIF referenciado durante el proceso EO.
1.e REPORTAR_PACIENTES	PACIENTE, MEDICO, PADECIMIENTO
2.e REPORTAR_MEDICOS	MEDICO, ESPECIALIDAD

El total de FTR por transacción se resume en la siguiente tabla:

Funcionalidad	FTR	
EI		
1.a AGREGAR_PACIENTE		3
1.b ACTUALIZAR_PACIENTE		3
1.d ELIMINAR_PACIENTE		1
2.a AGREGAR_MEDICO		2
2.b ACTUALIZAR_MEDICO		2
2.c ELIMINAR_MEDICO		1
3.a AGREGAR_ HISTORIAMEDICA		1
3.b ACTUALIZAR_ HISTORIAMEDICA		1
3.d ELIMINAR_ HISTORIAMEDICA		1
EQ	**Ent**	**Sal**
1.c CONSULTAR_PACIENTE	1	3
2.c CONSULTAR_MEDICO	1	2
3.c CONSULTAR_ HISTORIAMEDICA	1	4
4.a CONSULTAR_ ESPECIALIDADES	1	1
4.b REPORTAR_ ESPECIALIDADES	1	1
EO		
1.e REPORTAR_PACIENTES		3
2.e REPORTAR_MEDICOS		2

Elementos de datos (DET)

Para cada una de las transacciones, se cuenta el número de DET a los que afecta:

DET para entradas externas

Para considerar los DET en cada uno de los procesos de entrada externa, se puede seguir las siguientes reglas:

Entrada externa	Campo no recursivo significativo al usuario mantenido en un ILF por un EI	Campo no introducido por el usuario, pero mantenido en un ILF por un EI	Implementación física.
1.a AGREGAR_PACIENTE			
ClavePaciente	Sí	No	No
Cuarto	Sí	No	No
FechaIngreso	Sí	No	No
ClaveMedico	Sí	No	No
ClavePadecimiento	Sí	No	No
NombrePaciente	Sí	No	No
Edad	Sí	No	No
EstadoCivil	Sí	No	No
Direccion	Sí	No	No
Telefono	Sí	No	No
Total de DET	10		
1.b ACTUALIZAR_PACIENTE			
Cuarto	Sí	No	No
FechaIngreso	Sí	No	No
ClaveMedico	Sí	No	No
ClavePadecimiento	Sí	No	No
NombrePaciente	Sí	No	No
Edad	Sí	No	No
EstadoCivil	Sí	No	No
Direccion	Sí	No	No
Telefono	Sí	No	No
Total de DET	9		
1.d ELIMINAR_PACIENTE			
ClavePaciente	Sí	No	No
Cuarto	Sí	No	No
FechaIngreso	Sí	No	No
ClaveMedico	Sí	No	No
ClavePadecimiento	Sí	No	No
NombrePaciente	Sí	No	No
Edad	Sí	No	No
EstadoCivil	Sí	No	No
Direccion	Sí	No	No
Telefono	Sí	No	No
Total de DET	10		
2.a AGREGAR_MEDICO			
ClaveMedico	Sí	No	No
NombreMedico	Sí	No	No
ClaveEspecialidad	Sí	No	No
Horario	Sí	No	No
Total de DET	4		

Entrada externa	Campo no recursivo significativo al usuario mantenido en un ILF por una EI	Campo no introducido por el usuario, pero mantenido en un ILF por una EI	Implementación física.
2.b ACTUALIZAR_MEDICO			
NombreMedico	Sí	No	No
ClaveEspecialidad	Sí	No	No
Horario	Sí	No	No
Total de DET	3		
2.c ELIMINAR_MEDICO			
ClaveMedico	Sí	No	No
NombreMedico	Sí	No	No
ClaveEspecilidad	Sí	No	No
Horario	Sí	No	No
Total de DET	4		
3.a AGREGAR_HISTORIAMEDICA			
ClavePaciente	Sí	No	No
Historia	Sí	No	No
Total de DET	2		
3.b ACTUALIZAR_ HISTORIAMEDICA			
Historia	Sí	No	No
Total de DET	1		
3.d ELIMINAR_HISTORIAMEDICA			
ClavePaciente	Sí	No	No
Historia	Sí	No	No
Total de DET	2		

DET para consultas externas

Para contar los DET en cada uno de los procesos de consulta externa, es necesario considerar ambos lados de la consulta. Primero el lado de entrada de las consultas y posteriormente el lado de salida.

Entrada Consulta externa	**El campo especifica que debe ejecutarse una consulta. Contar como 1 DET**	**El campo especifica un criterio de selección de datos. Contar como 1 DET**	**Implementación física. Contar 1 DET**
1.b CONSULTAR_PACIENTE *ClavePaciente*	No	Sí	
Total de DET	1		
2.c CONSULTAR_MEDICO *ClaveMedico*	No	Sí	
Total de DET	1		
3.c CONSULTAR_ HISTORIAMEDICA *ClavePaciente*	No	Sí	
Total de DET	1		
4.a CONSULTAR_ ESPECIALIDADES *ClaveEspecialidad*	No	Sí	
Total de DET	1		
4.b REPORTAR_ ESPECIALIDADES *ClaveEspecialidad*	No	Sí	
Total de DET	1		

Salida **Consulta externa**	**Campo no recursivo significativo al usuario. Contar 1 DET**	**Campo literal, variable de paginación o stamp. No contar**	**Implementación física. Contar el grupo como 1 DET**
1.4b CONSULTAR_PACIENTE			
Cuarto	Sí		
FechaIngreso	–		Sí
NombreMedico	Sí		
Padecimiento	Sí		
NombrePaciente	Sí		
Edad	Sí		
EstadoCivil	Sí		
Direccion	Sí		
Telefono	Sí		
Total de DET	8		1
2.c CONSULTAR_MEDICO			
NombreMedico	Sí		
Especialidad	Sí		
Horario	—		Sí
Total de DET	2		1

Salida **Consulta externa**	**Campo no recursivo significativo al usuario. Contar 1 DET**	**Campo literal, variable de paginación o stamp. No contar**	**Implementación física. Contar el grupo como 1 DET**
3.c CONSULTAR_ HISTORIAMEDICA			
NombrePaciente	Sí		
NombreMedico	Sí		
NombrePadecimiento	Sí		
Historia	Sí		
Total de DET	4		
4.a CONSULTAR_ ESPECIALIDADES			
ClaveEspecialidad	Sí		
Especialidad	Sí		
Total de DET	2		
4.b REPORTAR_ ESPECIALIDADES			
ClaveEspecialidad	Sí		
Especialidad	Sí		
Total de DET	2		

DET para salidas externas

Para contar los DET en cada uno de los procesos de salida externa se puede revisar cada una de las reglas con la siguiente tabla:

Salida externa	Campo no recursivo significativo al usuario. Contar 1 DET	Campo literal, variable de paginación o etiquetas. No contar	Implementación física. Contar el grupo como 1 DET
1.e REPORTAR_PACIENTES			
Cuarto	Sí	No	No
FechaIngreso	--	--	Sí
NombreMedico	Sí	No	No
Padecimiento	Sí	No	No
NombrePaciente	Sí	No	No
Edad	Sí	No	No
EstadoCivil	Sí	No	No
Direccion	Sí	No	No
Telefono	Sí	No	No
TotalPacientes	Sí	No	No
Total DET	9		1
2.e REPORTAR_MEDICOS			
NombreMedico	Sí	No	No
Especialidad	Sí	No	No
Horario	No	No	Sí
TotalMedicos	Sí	No	No
Total DET	3		1

Ya contados los elementos para definir la complejidad de las funcionalidades de datos y de las transacciones, se hace un resumen indicando la complejidad.

Funcionalidad	RET/FTR		DET		Complejidad Funcional	
ILF						
PACIENTE	2		10		Baja	
MEDICO	1		4		Baja	
HISTORIA MEDICA	1		3		Baja	
EIF						
ESPECIALIDADES	1		4		Baja	
PADECIMIENTOS	1		2		Baja	
EI						
1.a AGREGAR_PACIENTE	3		10		Alta	
1.b ACTUALIZAR_PACIENTE	3		9		Alta	
1.d ELIMINAR_PACIENTE	1		10		Baja	
2.a AGREGAR_MEDICO	2		4		Baja	
2.b ACTUALIZAR_MEDICO	2		3		Baja	
2.c ELIMINAR_MEDICO	1		4		Baja	
3.a AGREGAR_HISTORIAMEDICA	1		2		Baja	
3.b ACTUALIZAR_HISTORIAMEDICA	1		1		Baja	
3.d ELIMINAR_HISTORIAMEDICA	1		2		Baja	
EQ	**Ent**	**Sal**	**Ent**	**Sal**	**Ent**	**Sal**
1.b CONSULTAR_PACIENTE	1	3	1	9	Baja	Promedio
2.c CONSULTAR_MEDICO	1	2	1	3	Baja	Baja
3.c CONSULTAR_HISTORIAMEDICA	1	4	1	4	Baja	Promedio
4.a CONSUTAR_ESPECIALIDADES	1	1	1	2	Baja	Baja
4.b REPORTAR_ESPECIALIDADES	1	1	1	2	Baja	Baja
EO						
1.e REPORTAR_PACIENTES	3		10		Promedio	
2.e REPORTAR_MEDICOS	2		4		Baja	

Paso 5. Determinar el cálculo de puntos de función sin ajustar.

La tabla siguiente da el total de todas las funcionalidades para determinar los puntos de función sin ajustar (UFP).

Funcionalidad	Complejidad funcional		Totales de complejidad	Totales por Funcionalidad
ILF	3	Bajo	x 7 = 21	
	0	Promedio	x 10 = 0	
	0	Alto	x 15 = 0	21
EIF	2	Bajo	x 5 = 10	
	0	Promedio	x 7 = 0	
	0	Alto	x 10 = 0	10
EI	7	Bajo	x 3 = 21	
	0	Promedio	x 4 = 0	
	2	Alto	x 6 = 12	33
EQ	3	Bajo	x 3 = 9	
	2	Promedio	x 4 = 8	
	0	Alto	x 6 = 0	17
EO	1	Bajo	x 4 = 4	
	1	Promedio	x 5 = 5	
	0	Alto	x 7 = 0	9
Suma de Puntos de Función sin Ajustar (UFP) =				**90**

Paso 6. Determinar el valor del factor de ajuste.

Para este paso se evalúan las 14 características generales del sistema. El punto 5 de los requerimientos permite evaluar cada una de ellas.

Característica	Grado de Influencia	Característica	Grado de influencia
1 - Comunicación entre datos	4	8 - Actualización en línea	1
2 - Procesamiento distribuido de datos	0	9 - Complejidad del proceso	0
3 - Objetivos de desempeño	0	10 - Reusabilidad	4
4 - Grado de configurabilidad	0	11 - Facilidad de instalación	0
5 - Volumen de transacción	0	12 - Facilidad de operación	1
6 - Captura en línea	5	13 - Múltiple instalación	1
7 - Eficiencia al usuario final	1	14 - Facilidad de cambio	2

Suma de los grados de influencia (TDI) = 19

Esto indica que el total grados de influencia es 16. Para calcular el valor del factor de ajuste (VAF):

VAF = 0.65 + (0.01 x 19) = 0.84

Paso 7. Calcular el total de puntos de función.

El total de puntos de función queda entonces:

FP = UFP x VAF = 90 x 0.84 = **75.6**

Para dar una idea de lo que esto significa en términos de esfuerzo, la siguiente tabla 4.1 muestra los promedios en la industria del software en Estados Unidos.

Tamaño del proyecto (Puntos de función)	Productividad (Puntos de función / mes) *	Tamaño del equipo recomendado +
5	5.6	1
10	6.6	1
20	7.8	1
40	9.2	1
80	9.8	1.35
160	8	2
320	5.8	4

640	4.3	8
1,280	3.1	27

* Fuente: Applied Software Measurement, Capers Jones. McGraw Hill, 1995

+ Fuente: Applied Software Measurement, Capers Jones. McGraw Hill, 1991

Tabla 4.1 Promedios en la industria de los Estados Unidos.

CAPÍTULO V
OTROS DESARROLLOS DE LA METODOLOGÍA DEL ANÁLISIS DE PUNTOS DE FUNCIÓN

Como se observó en el capítulo I, la métrica de análisis de puntos de función tiene aproximadamente 20 años de creada y la última modificación a las reglas y definiciones estándares descrita por IFPUG en el *Function Point Counting Practices Manual*, es la versión 4.0 publicada en 1994, es decir, hace 5 años. Durante todo este tiempo la cantidad de usuarios se ha incrementado de manera importante y también el mundo de la tecnología. Es importante entonces, adaptar el análisis de puntos de función a nuevas tecnologías.

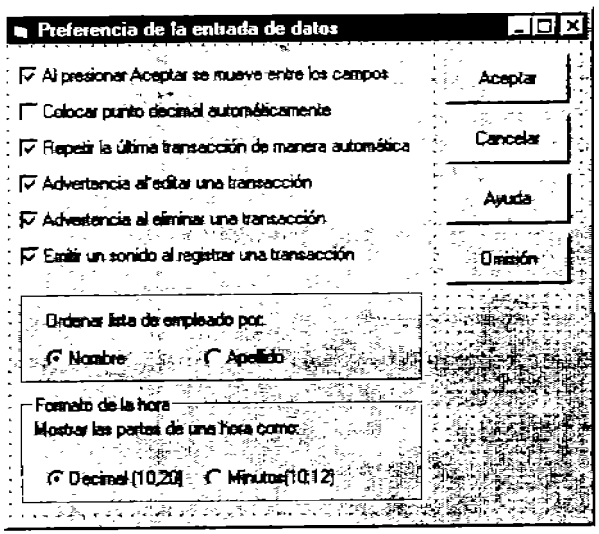
El propósito de este capítulo es mostrar un panorama de lo que se ha escrito con respecto a la aplicación de puntos de función a nuevas tecnologías y mostrar las variantes que sobre puntos de función existen.

5.1 APLICACIONES GUI - INTERFAZ GRÁFICA DE USUARIO

En aplicaciones con interfaz de usuario, el manual de IFPUG no proporciona mucho detalle de cómo contar botones tipo radio (Radio Buttons), listas de verificación (Check Boxes), listas de elección (Pick List), cajas tipo combo (Combo Boxes), etc. Las consultas externas son muy comunes en este tipo de aplicaciones. Un EQ no muestra datos calculados. Una lista dinámica (drop down box) es un ejemplo de una consulta externa. Ya que simplemente lee de un archivo. Por ejemplo, una lista como la anterior, debe contarse como una consulta externa (EQ) si los nombres de los países se encuentran en un archivo lógico interno o un archivo de interfaz externa; la selección de uno de los elementos desplegados al apuntar y dar clic, es un DET.

Los botones tipo radio (Radio Buttons) son tratados como tipos de elementos de datos (DET). Dentro de un grupo de botones el usuario tiene la opción de elegir sólo un botón a la vez. Así, se cuenta solamente un DET para todos los botones que se encuentran dentro de un marco (frame).

En el caso de las cajas de verificación (Check Boxes) cada una de las cajas representan un DET. Pues cada una es independiente de las otras y dan una funcionalidad diferente a la aplicación. Por lo tanto la siguiente pantalla contiene 9 DET, (8 DET y una tecla de acción).



Los botones de comando (Command Buttons) pueden especificar una acción de agregar, cambiar, eliminar o consultar. De acuerdo con las reglas IFPUG cada botón de este tipo debe ser contado como un DET por la acción que invoca. Por ejemplo, si al oprimir un botón de comando "Agregar", este almacena en un archivo el nombre, dirección, ciudad, estado, código postal, teléfono y fax de ciertos empleados a un archivo lógico interno (ILF). esto representaría 7 DET de los campos, más el botón, serían 8 DET. Es decir, el botón "Agregar" es una entrada externa con 8 DET. Lo mismo un botón "Cambiar" y lo mismo para un botón de "Eliminar".

Las funciones de usuario como son crear, actualizar o eliminar, deben contarse como entradas externas; pero hay que asegurar que los datos son para mantenimiento y no sólo como pantallas de visualización. Por ejemplo, elegir un elemento de una lista de elección (pick-list) por lo generar es un DET en una entrada externa y no una entrada externa por separado.

Las funciones de usuario que tienen que ver con cálculo o búsqueda de datos en un archivo lógico interno (ILF) para ser desplegado, deben contarse como salidas externas. Por ejemplo un "Abrir Archivo" con la lista de archivos que recién se abrieron.

Las funciones que automáticamente proporciona Windows o el sistema operativo y no la aplicación no deben contarse; por ejemplo, minimizar y maximizar la ventana, etc.

Menús, iconos, barras de recorrido (scroll bars) y otros dispositivos de navegación no deben contarse a menos que se utilicen para recuperar datos. Una barra de acción, un botón, etc., pueden ser el lado de entrada de un consulta externa (EQ).

El siguiente ejemplo muestra como contar puntos de función en una aplicación ficticia, con interfaz de usuario.

Suponga que la figura anterior es el prototipo de una pantalla de captura que va a ser utilizada por una empresa de mensajería. Esta pantalla de información es desplegada al momento de capturar el número de orden descrito en la parte superior.

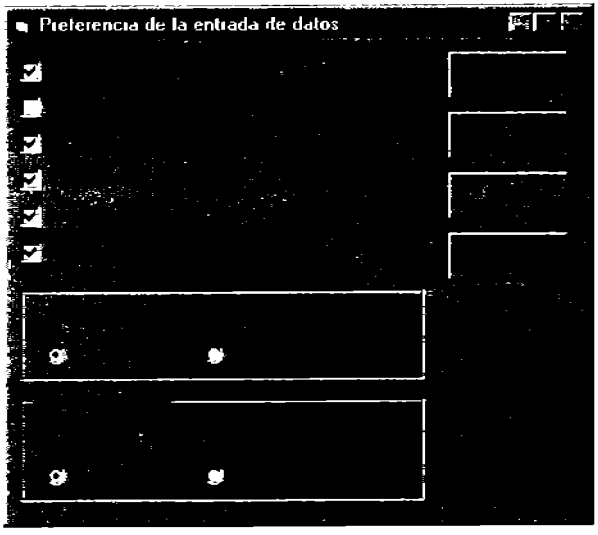
Paso 1. Determinar el tipo de aplicación a contar.

El primer paso es saber el tipo de aplicación a contar. Vamos a suponer que es la primera vez que se cuenta este sistema, así haremos el análisis de toda la aplicación.

Paso 2. Identificar la frontera de la aplicación.

En este ejemplo, supondremos que todos los archivos que se utilizan en esta aplicación están representados en esta pantalla.

Los datos de esta pantalla son tomados de dos archivos lógicos internos, el archivo "orden" y el archivo "agente". Todos los campos que pueden capturarse son los de el archivo "orden" y el archivo "agente" sólo se utiliza para búsquedas y validación de los números de agente y no se puede mantener desde esta aplicación, lo cuál significa que se encuentra fuera de la frontera de la aplicación.



Paso 3. Determinar la funcionalidad de datos. Archivos lógicos internos y de interfaz externa.

Como el archivo "orden" es actualizado con la aplicación, lo contamos como un archivo lógico interno (ILF); como el archivo "agente" sólo se consulta pero no se actualiza, lo contamos como un archivo de interfaz externa (EIF). Para determinar su complejidad, tomamos dos factores, el número de elementos de registro (RET) y el número de elementos datos (DET).

Cada uno de estos archivos tiene un RET. Un ejemplo de varios RET sería en el caso de que existieran diferentes tipos de orden que mostraran otra información. Los DET son esencialmente los campos de datos que existen dentro del archivo. Campos que se repiten como el de los números telefónicos no se cuentan por separado. Sólo se cuenta un DET por tipo de campo número de teléfono, aún si aparece muchas veces. Entonces en este caso la complejidad queda:

Nombre de archivo	Funcionalidad	RET	DET	Complejidad
"Orden"	ILF	1	37	Baja
"Agente"	EIF	1	2	Baja

Paso 4. Determinar la funcionalidad de transacciones. Entradas, salidas y consultas externas.

Vamos a determinar el número de entradas, salidas y consultas asociadas con esta pantalla. La primera acción que se presenta al desplegarse esta pantalla es una consulta externa (EQ) que muestra los datos a ser editados. La complejidad de una EQ se determina contando el número de archivos referenciados (FTR) y el número de elementos de datos (DET). Esta consulta externa despliega datos de dos archivos lógicos y presenta 39 DET. Hay que recordar que esta complejidad se determina mediante los componentes de entrada y salida, eligiendo la mayor (por lo general es el componente de salida).

Al presionar Aceptar se inicia una transacción de entrada (EI) para actualizar el archivo "orden". Para saber la complejidad de esta transacción se utiliza el mismo criterio que para el EQ. Esta entrada externa actualiza un archivo referenciado, el archivo "orden" y los campos actualizables son 37. Se cuenta además un DET por la acción de oprimir el botón del ratón que inicia la transacción. Por lo que son 38 DET. Si existiera en esta pantalla la posibilidad de eliminar o agregar órdenes, se contarían como transacciones de entrada separadas.

Los mensajes de error asociados con una entrada se cuentan como una salida separada. Si suponemos que existe un mensaje de error al intentar introducir un código de agente que no existe y que existen otros dos mensajes de error que avisan que se requiere introducir obligatoriamente las direcciones del origen y del destino, entonces contamos un archivo referenciado (FTR) para la validación de código de agentes, el archivo "agente" y un elemento de dato (DET) para cada uno de los distintos mensajes, en este caso 3.

Resumiendo las transacciones de este paso nos queda la siguiente información:

Proceso	Funcionalidad	FTR	DET	Complejidad
Despliegue	EQ	2	39	Alta
Editar	EI	1	38	Promedio
Mensajes de error	EO	1	3	Baja

Paso 5. Determinar los tipos de función sin ajustar (UFP)

Aplicando los factores de peso, tenemos lo siguiente:

Funcionalidad	Baja	Promedio	Alta	
ILF	0 x 7	1 x 10	0 x 15	= 10
EIF	1 x 5	0 x 7	1 x 10	= 15
EI	0 x 3	1 x 4	1 x 6	= 4
EQ	0 x 3	0 x 4	1 x 6	= 6
EO	1 x 4	0 x 7	0 x 10	= 4

Por lo tanto, la suma de puntos de función sin ajustar (UFP) es de 39.

Paso 6. Determinar el valor del factor de ajuste (VAF) con las 14 características generales del sistema de acuerdo a su grado de influencia. Vamos a suponer que la suma de los grados de influencia de estas características dan un total de 15. El valor del factor de ajuste es:

VAF = 0.65 + (0.01 x 15) = 0.8

Paso 7. Por último calcular el total de puntos de función (TFP) que es el producto de UFP y VAF.

TFP = 39 x 0.8 = 31.2

5.2 APLICACIONES EN LAS PÁGINAS WEB DE INTERNET

Anteriormente las páginas web contenían poco o nada de puntos de función, pues sólo eran pantallas de presentación donde no se mantenía a ningún archivo lógico y lo que se mostraba no era resultado de algún archivo leído; es decir, la mayoría de las páginas Internet eran referencias o ligas a otras páginas, lo que en nuestro lenguaje le llamamos "navegar". Sin embargo, el panorama en Internet es diferente pues existen cada vez más páginas con funcionalidad desde la perspectiva de puntos de función; por ejemplo, de empresas que anuncian o venden productos por este medio o las páginas de los bancos donde los clientes realizan transacciones. Estas son las aplicaciones que necesitan ser medidas.

Aplicar puntos de función a este tipo de aplicaciones tiene sus problemas. El primer problema y el más difícil es identificar la frontera de la aplicación. El segundo problema es identificar transacciones, archivos lógicos internos y/o de interfaz externa.

La frontera en las aplicaciones de internet, al igual que en las tradicionales, se identifica tomando en cuenta toda la aplicación. La pregunta para poder identificar la frontera es: ¿qué es lo que va a mantener cada una de las páginas html de la aplicación que estamos contando? Seguramente los archivos de interfaz externa no son mantenidos por las páginas, pero sí por los propietarios del servidor donde reside la aplicacion internet, mediante otra aplicación.

En cuanto a la funcionalidad de transacción (entradas, salidas y consultas externas) se siguen también la mismas reglas básicas que en las aplicaciones tradicionales. Un ejemplo de una entrada externa en una aplicación Internet es una forma de captura. Estas formas pueden ser un archivo html que se actualiza con información. Las ligas que son solamente para navegar, no son contadas en puntos de función.

5.3 PUNTOS DE FUNCIÓN EN CASOS DE USO

Los puntos de función miden el tamaño de los sistemas de información de acuerdo con lo que el usuario ve e interactúa. Un "caso de uso" es un diálogo entre un usuario y una computadora con el fin de satisfacer necesidades específicas del usuario.

Por lo general, una pantalla necesita ser separada en transacciones (entradas, salidas y consultas externas). Es decir, cualquier pantalla particular puede tener un EQ (despliegue de información), EO (datos calculados) y EI (actualización de archivos) . Los casos de uso también son partes (pasos) separadas de un sistema. Un paso puede ser una transacción, un elemento de dato o ninguna de las dos. Cada paso necesita ser analizado para determinar si es una transacción o elemento de dato.

Dekkers en [11] hace mención a lo siguiente en relación a como contar puntos de función para casos de uso. "*Los casos de uso presentan los requerimientos lógicos del usuario en un formato completo y fácilmente digerible que ayuda a contar puntos de función de la manera más directa. Los modelos de objetos proporcionan un mapeo de objetos y clases independientes y dependientes. Aquellos objetos (o grupos de objetos dependientes) mantenidos a través de servicios estándar (descritos por uno o más casos de uso) se cuentan, por lo general, como Archivos lógicos internos (ILFs) . Objetos y clases que solamente son referenciados y no mantenidos a través de funciones de estándar se cuentan como Archivo de interfaz externa (EIFs). Los mismos casos de uso dan la información funcional y de proceso necesario para contar los componentes restantes de los puntos de función.*"

5.4 VARIANTES DEL ANÁLISIS DE PUNTOS DE FUNCIÓN

En esta sección mostraremos algunas de las variantes para el análisis de puntos de función, que se han propuesto para dar alternativas a la medida final. Algunos dan más ponderación a algunos aspectos de los sistemas, como por ejemplo, la construcción de algoritmos.

Puntos de Función Mark II

A principios de 1980, Charles R. Symons, un investigador de la compañía Nolan, Norton en Londres, desarrolló un esquema para contar puntos de función llamado **Mark II**. Entre otras modificaciones, Mark II extiende el número de factores de ajuste de 14 a 19 con la idea de premiar no solo el valor y tamaño de una métrica sino también el esfuerzo. Por lo que los factores añadidos en Mark II fueron los siguientes:

- Software con mayor interfaz de sistema
- Software con consideraciones de alta seguridad
- Software que proporciona acceso directo a terceras partes.
- Software con requerimientos de documentación especial.
- Software que necesita entrenamiento especial a los usuarios.
- Software que necesita hardware especial.

Al contar con esta métrica una misma aplicación que con puntos de función, ésta difiere aproximadamente en mas de un 30%.

Método "Backfire" de SPR

Para esta métrica también creada por SPR en 1985, se debe conocer el lenguaje de programación y la cantidad de instrucciones de código fuente que se utilizaron para implementar la aplicación. Para obtener los puntos de función de una aplicación se utiliza una tabla que relaciona la cantidad de instrucciones por punto de función de diversos lenguajes. Por ejemplo, un programa en Turbo C++ que según la tabla es nivel 6, tiene un promedio de 23 instrucciones por punto de función. Si este programa cuenta con 200 instrucciones, una aproximación en puntos de función sería:
200 instrucciones / 23 instrucciones por puntos de función = 8.69 puntos de función.

Método de Feature Points de SPR

Otra adaptación, llamada **punto-característica (feature points)** es un superconjunto de la métrica de IFPUG donde se considera el número de algoritmos en el programa además de los cinco componentes estándar para el cálculo de puntos de función. Este método también reduce el peso de los ILF de 10 (promedio) a 7. La diferencia principal entre ambas metodologías es la forma de calcular la complejidad. Mientras que en IFPUG se calcula mediante la influencia de las 14 características generales del sistema y la tabla de pesos para cada uno de los componentes, en la técnica SPR la complejidad se define mediante dos preguntas descritas a continuación con sus pesos asociados:

- ¿Complejidad del problema?
  1. Algoritmos y cálculos simples.
  2. La mayoría de los algoritmos y los cálculos son simples.
  3. Los algoritmos y los cálculos tienen una complejidad promedio
  4. Algunos cálculos son difíciles o complejos.
  5. Muchos de los algoritmos y cálculos son muy complejos.
- ¿ Complejidad en los datos?
  1. Datos simples con pocas variables y de baja complejidad.
  2. Diversas variables pero las relaciones entre los datos son simples.
  3. Múltiples archivos, campos e interacción entre los datos.
  4. Estructuras de archivo complejas e interacciones entre los datos.
  5. Estructuras de archivo muy complejas e interacción entre los datos.

Sumando el peso de las preguntas da el factor de complejidad según la siguiente tabla:

Suma de la complejidad del programa y de los datos	Multiplicador de complejidad
2	0.6
3	0.7
4	0.8
5	0.9
6	1
7	1.1
8	1.2
9	1.3
10	1.4

Este factor de complejidad se aplica al total de puntos de función sin ajustar dada en la siguiente tabla, donde observamos que se añade el componente de número de algoritmos y vemos que no es necesario contar DET, RET o FTR.

Parámetro	**Peso**	**Total**
Número de algoritmos	x 1	=
Número de entradas	x 4	=
Número de salidas	x 5	=
Número de consultas	x 4	=
Número de archivos de datos	x 10	=
Número de interfaces	x 7	=
Puntos de función sin ajustar		=

La complejidad por los puntos de función sin ajustar dan el total de puntos de función.

Full Function Points

Otra adaptación importante desarrollada en el *Software Engineering Laboratory in Applied Metrics (SELAM)* de la Universidad de Quebec en Montreal, llamada **Puntos de función completos (Full Function Points),** considera contar puntos de función para software de tiempo real. Cabe señalar que este grupo participa de manera muy intensa con IFPUG. Con frecuencia aportan casos de estudio y artículos con nuevas innovaciones a los puntos de función.

El método de aproximación SPR de 1990

SPR enfocó el problema a intentar crear un punto de función preciso en la ausencia de un conocimiento completo de una aplicación. Aunque en primera instancia esto parece imposible, el método de aproximación tiene un mérito considerable y tiene características muy poderosas. Este método se basa en descubrir y utilizar relaciones entre los factores que son conocidos con precisión y los factores "ocultos" que no se conocen con precisión. Se basa en observar cuántas entradas, salidas, consultas, archivos e interfaces desde diversos tipos de software. Por ejemplo, al considerar un tipo de aplicación como un pago de nómina, puede suceder un patrón como el siguiente:

Factor	Porcentaje de Funcionalidad
Entradas	30
Salidas	20
Consultas	5
Archivos lógicos	40
Interfaces	5
Total	100

Con esta información esta métrica termina por concluir un valor de puntos de función.

Además de las anteriores, existen diversas variaciones de la técnica como son los puntos de función 3-D, puntos de función de ingeniería (engineering function points) y puntos objeto (object points). La Organización Internacional de Estándares (ISO) junto con los principales grupos de usuarios de puntos de función se encuentran trabajando para consolidar un conjunto de reglas que involucre todas estas nuevas ideas sobre puntos de función.

CAPÍTULO VI
CONCLUSIÓN

El análisis de puntos de puntos de función para medir el tamaño del software es una metodología que, para poder aplicarla requiere de cierta capacitación y experiencia en la misma. Por lo general, esta tarea la realizan especialistas que han aprobado un examen de certificación, pero lo más importante es tener mucha práctica con un cierto número de proyectos analizados.

En México, la mayoría de las universidades y escuelas superiores ya comienzan a incluir en sus planes de estudio temas relacionados con la Ingeniería de Software, sobretodo con temas referentes a la calidad de los productos de software. En mi opinión es necesario que estos cursos incluyan temas sobre las distintas métricas de software, ya que estas son necesarias para asegurar la calidad.

Las empresas dedicadas al desarrollo de software interesadas en aplicar estos conceptos dentro de su proceso de producción, deben reunir gente experta en ingeniería de software y capacitarlas para desarrollar e implementar procesos que permitan medir y mejorar el desarrollo de software.

Ante la necesidad de las empresas de poder contar con un estándar para medir el tamaño del software, se debe tener conocimiento de pros y contras de las distintas métricas incluyendo la de puntos de función. Es necesario tomar en cuenta que la métrica debe ser lo suficientemente sencilla para poder ser aplicada por el equipo de trabajo del proyecto.

Aunque el análisis de puntos de función no es una métrica perfecta, su uso se ha extendido en todo el mundo, y cada vez se discute más a fondo la metodología tratando de mejorarla. Entre los mayores inconvenientes que se cuestionan son:

- Los pesos asociados con la complejidad parecen muy subjetivos, sin embargo, la experiencia de los usuarios continúa ajustando estas cantidades.
- Esta enfocado a sistemas de información de negocio, por ejemplo, bancos. Lo cuál es cierto, aunque cada vez existen más artículos que con la misma filosofía, adaptan esta metodología a nuevos tipos de software como son aplicaciones Internet, OO, etc.

- Es una tarea muy intensa que requiere de esfuerzo tanto para aprenderlo como para aplicarlo, sin embargo, poco a poco irán saliendo al mercado herramientas automatizadas que ayuden a facilitar el conteo.

En México, debemos aprovechar el auge que está teniendo esta metodología y tratar de adoptarla al tipo de proyectos que se realizan en nuestro medio, acordando estándares adecuados. Es necesario comenzar a preparar gente especialista a fin de difundirla en todo el país.

A pesar de los pros y contras que tiene está métrica, no cabe duda que es la que más aceptación ha tenido en la industria del software a nivel mundial y por lo tanto ya ha sido adoptada como la métrica estándar en muchas empresas en el mundo.

Por último, en mayo de 1999, durante el transcurso de la realización de esta tesis, se liberó la versión 4.1 del manual de prácticas de conteo de IFPUG.

ANEXO

FORMAS DE AYUDA PARA EL ANÁLISIS DE PUNTOS DE FUNCIÓN

ANÁLISIS DE PUNTOS DE FUNCIÓN RESUMEN POR TIPO DE FUNCIÓN			
Descripción	RETs/FTRs	DETs	Complejidad
ILF			
EIF			
EI			
EQ			
EO			

TABLA PARA EL CÁLCULO DE PUNTOS DE FUNCIÓN

Número de proyecto: _____ Nombre del proyecto: _____

Fase proyecto: Requerimientos / Diseño / Construcción / Prueba /Entrega (subraye uno)

Fecha del cálculo: _____ Nombre del contador: _____

Funcionalidad	Bajo	Promedio	Alto	Total
Archivos lógicos internos	___x 7	___x 10	___x 15	= ____
Archivos de interfaz externa	___x 5	___x 7	___x 10	= ____
Entradas externas	___x 3	___x 4	___x 6	= ____
Salidas externas	___x 4	___x 5	___x 7	= ____
Consultas externas	___x 3	___x 4	___x 6	= ____

Suma de puntos de función sin ajustar (UFP) =_____

14 Características Generales del Sistema

Característica	Grado de influencia	Característica	Grado de influencia
1 - Comunicación entre datos	_____	8 - Actualización en línea	_____
2 - Funciones distribuidas	_____	9 - Complejidad del proceso	_____
3 - Desempeño	_____	10 - Reusabilidad	_____
4 - Configurable	_____	11 - Facilidad de instalación	_____
5 - Volumen de transacción	_____	12 - Facilidad de operación	_____
6 - Captura en línea	_____	13 - Múltiples sites	_____
7 - Eficiencia al usuario final	_____	14 - Facilidad de cambio	_____

Suma de los grados de influencia (TDI) = _____

VAF = Valor del factor de ajuste = 0.65 + (0.01 x TDI) = _____

FP = Total de puntos de función = UFP X VAF = _____

GLOSARIO

Actualizar. El hecho de modificar datos.

Aplicación. Conjunto de uno o más componentes, módulos o subsistemas. Con frecuencia se usa como sinónimo de sistema.

Archivo lógico. Para tipos de función de datos, un grupo de datos relacionado de manera lógica y no la implementación física. "Archivo de empleado" y no "h900102".

Datos calculados. Datos que requieren de un proceso distinto al de una recuperación directa o la edición de información desde un archivo lógico interno (ILF) o un archivo de interfaz externa (EIF).

Datos de control. Datos utilizados por la aplicación para controlar, directa o indirectamente, el comportamiento de una aplicación.

Defecto. Es una desviación de la especificación del sistema[1]. Un defecto se presenta cuando el software no se comporta de la manera que el usuario espera que lo haga.[2]

Desarrollo. Especificación, construcción y entrada de un nuevo sistema de información.

Frontera. Línea divisoria entre la aplicación por contar y otras aplicaciones, o el dominio del usuario.

IFPUG. Iniciales de "International Function Points Users Group", organismo encargado de establecer estándares y promover el uso de puntos de función.

---

[1] Hetzel, Bill
[2] Myers, Glen

Información de control. Son datos utilizados por un proceso que aseguran el cumplimiento con los requerimientos de la funcionalidad del negocio especificados por el usuario. No es necesario que mantengan un archivo lógico interno. Por ejemplo, un mensaje que se despliega advirtiendo un error o que se requiere de alguna acción.

Mantener. Se refiere a la habilidad de modificar los datos mediante un proceso elemental. Es decir, al hecho poder agregar, cambiar y eliminar datos.

Mejora. Modificación de un sistema existente.

Perspectiva funcional. Punto de vista de la funcionalidad dada por la aplicación; excluye consideraciones técnicas y de implementación.

Proceso. Un conjunto de operaciones o actividades que actúa en entradas de información para producir un resultado.

Proceso elemental. Es la unidad de actividad más pequeña que es significativa al usuario final en el negocio.

Significativo al usuario. Se refiere a los requerimientos específicos que un usuario experimentado definiría para la aplicación.

Funcionalidad de datos. La funcionalidad proporcionada al usuario a fin de que coincida con los requerimientos de datos. Estos son los archivos lógicos internos (ILF) y los archivos de interfaz externa (EIF).

Funcionalidad de transacciones. La funcionalidad proporcionada al usuario por una aplicación para procesar datos. Los tipos de función de transacción son entradas externas (EI), salidas externas (EO) y consultas externas (EQ).

Transacción. Todos los procesos asociados con una ocurrencia de un disparo externo. Por ejemplo, en un reloj, cada tic del pantalla del tiempo es un disparo. Todos los procesos asociados con cada nuevo tic es una transacción separada.

Usuario. Un ser humano o aplicación que interactúa con la aplicación a ser medida.

Bibliografía

[1] A.J. Albrecht, "Measuring Application Development Productivity", *Proc IBM Applications Development Symp.*, Monterey, Calif., Oct 14-17, 1979.

[2] A.J. Albrecht y J.E. Gaffney, "Software Functions, Source Lines of Code, and Development Effort Prediction: A Software Science Validation", *IEEE Trans. Software Eng.*, vol. 9, no. 6, Nov, 1983.

[3] IBM CISGuidelines 313, *AD/M Productivity Measurement and Estimation Validation*, Nov. 1984.

[4] International Function Point Users Group (IFPUG), *Function Point Counting Practices Manual, Release 4.0*, IFPUG, Westerville, Ohio, Enero 1994.

[5] Garmus, David y Herron, David, *Measuring the Software Process, a practical guide to funciontal measurements*, Yourdon Press Computing Series, 1996.

[9] David H. Longstreet, The Longstreet Consulting Inc. *Function Points Applied to New and Emerging Tecnologies*, 1999. Este artículo se encuentra en la dirección web *http://www.SoftwareMetrics.Com/Articles/index.htm*

[6] Capers, Jones, *Sizing Up Software*, Scientific American: Feature Article, Diciembre, 1998. *http://www.sciam.com/1998/1298issue/1298jones.html*

[11] Capers, Jones, *Applied Software Measurement. Assuring Productivity and Quality*, 2a. edición. Ed. McGraw-Hill, 1997. ISBN 0-07-032826-9

[7] Dekkers, Carol, Quality Plus Technologies, *Use Cases and Function Points — Where's the Fit?*, publicación pendiente en IT Metrics Strategies, Diciembre, 1998.

[8] Abran, Alain y Nguyen, Tho-Hau, Fetcke, T, *Mappiing the OO-Jacobson Approach into Function Point Analysis*, en la 6th International Workshop on Software Metrics, F. Lehner, Regensburg, Alemania, 1997.

[9] International Function Point Users Group (IFPUG). *Function Point Counting Practices: Case Study 1. Release 1.0*, IFPUG, Westerville, Ohio, Julio 1994.