

Estimación de Tamaño de Software: Puntos de Función

CSOF-5101

- Rafael Meneses

- Introducción
- Historia
- Puntos de Función Vs. LOCs
- Cálculo de Puntos de Función (PF)
- Proceso de estimación mediante PF
- Referencias

- Introducción
- Historia
- Puntos de Función Vs. LOCs
- Cálculo de Puntos de Función (PF)
- Proceso de estimación mediante PF
- Referencias

- Métrica para cuantificar la funcionalidad de un sistema de software tal como la percibe el usuario/comprador
- Permite medir el tamaño del sistema a partir de sus requisitos iniciales
- Puede usarse muy tempranamente en el proceso de desarrollo ya que resulta relativamente fácil de visualizar

- Introducción
- **Historia**
- Puntos de Función Vs. LOCs
- Cálculo de Puntos de Función (PF)
- Proceso de estimación mediante PF
- Referencias

- Propuesta originalmente por A. Albrecht (IBM) en 1979
- En los 80s la técnica fue refinada y la organización IBM GUIDE produjo el primer manual de conteo
- Al finalizar los 80s se fundó *IFPUG - International Function Point Users*.
- En 1994, IFPUG produjo el release 4.0 de su *Manual de Prácticas de Conteo*
- El release 4.1 del manual se produjo en 1999

- Introducción
- Historia
- Puntos de Función Vs. LOCs
- Cálculo de Puntos de Función (PF)
- Proceso de estimación mediante PF
- Referencias

- Falta de una definición universalmente aceptada de lo que es exactamente una *Línea-de-Código (LOC)*
- Dependencia del lenguaje
- Dificultad en la estimación de LOCs que se necesitan para desarrollar un sistema a partir de información disponible en las fases de análisis y diseño
- Las líneas de código hacen énfasis en la fase de implementación, que es sólo una parte de la ejecución de un proyecto de desarrollo de software

¿Qué son Puntos de Función?

- Permiten medir el *tamaño del software* cuantificando la *funcionalidad proporcionada al usuario* basándose únicamente en un *diseño lógico* y en *especificaciones funcionales*
- *Análisis de Puntos de Función* → Método para cuantificar el tamaño y la complejidad de un sistema de software en términos de las funciones que el sistema entrega al usuario
- Método independiente de:
 - Lenguaje de programación
 - Proceso de desarrollo
 - Tecnología
 - Capacidad del equipo de desarrollo

- PFs son independientes del lenguaje, herramientas o tecnologías utilizadas
- PFs pueden ser estimados en etapas tempranas de análisis y diseño
- Debido a que los PFs están basados en la vista de un usuario externo al sistema, el personal no técnico tiene una mejor comprensión de lo que se está midiendo
- Con los PFs se puede tener una visión completa de las etapas de construcción del sistema y no solamente de su codificación

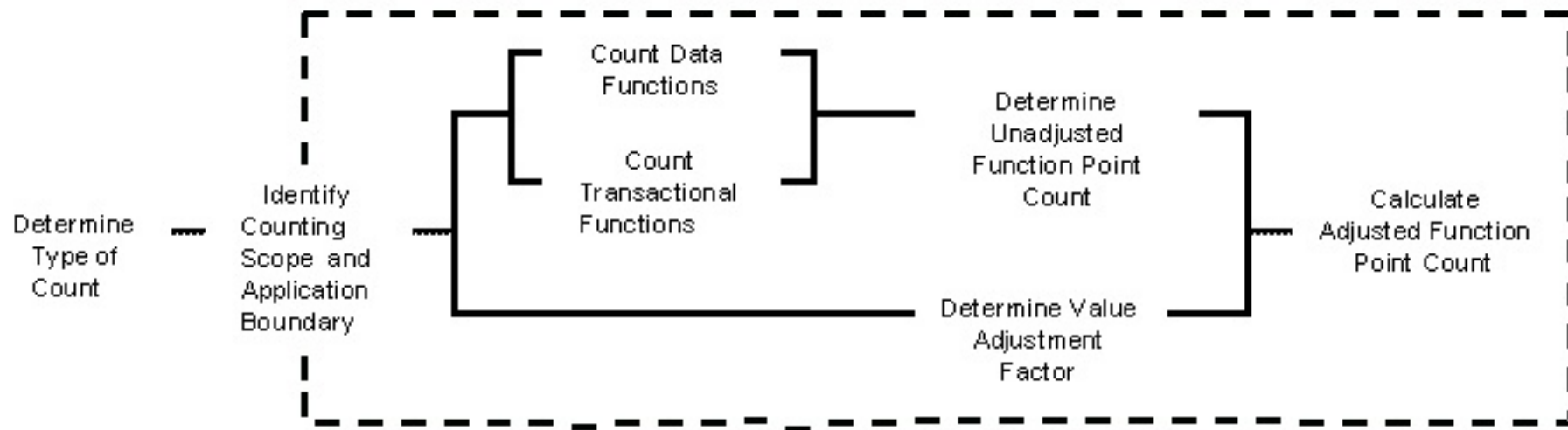
Uso de los Puntos de Función

- Medir productividad (p.e. # of PFs alcanzados x hora de trabajo invertida)
- Estimar desarrollo y soporte (p.e. análisis costo-beneficio, estimación de personal)
- Monitorear contratos de outsourcing
- Manejar cambios relacionados con decisiones de negocio
- Normalizar otras métricas (p.e. calcular defectos, normalmente requiere el tamaño en puntos de función)

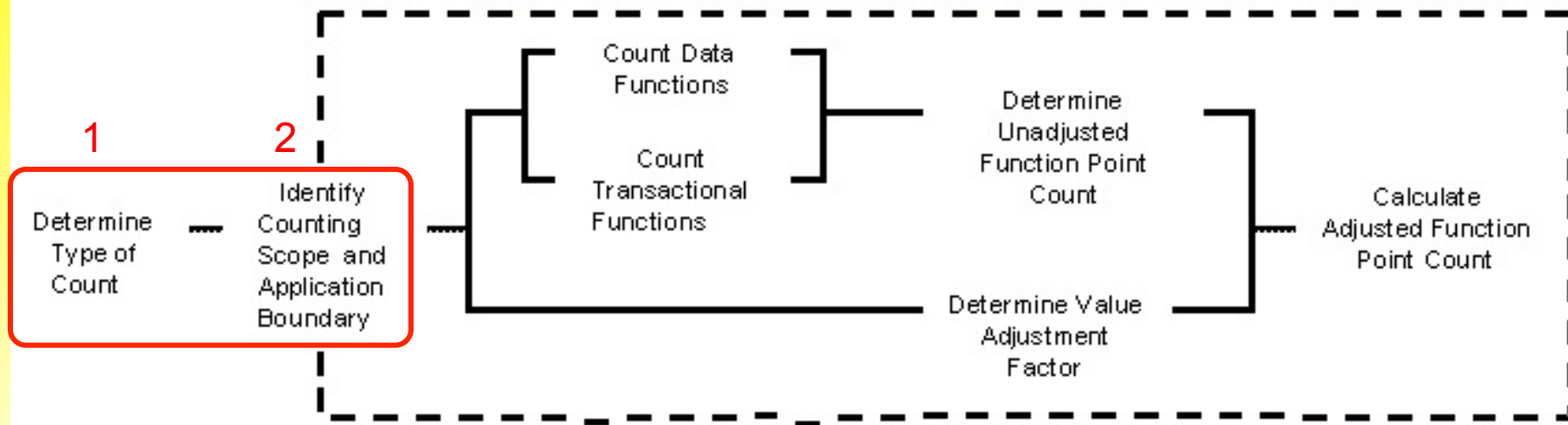
- Introducción
- Historia
- Puntos de Función Vs. LOCs
- Cálculo de Puntos de Función (PF)
- Proceso de estimación mediante PF
- Referencias

1. Determinar el *tipo* de conteo de puntos de función
2. Identificar el *alcance del conteo* y la *frontera de la aplicación*
3. Determinar los *puntos de función sin ajuste*
 - Calcular las funciones de datos
 - Calcular las funciones de transacción
4. Determinar el *factor de complejidad técnica* para ajuste
5. Calcular los *puntos de función ajustados*

Procedure Diagram

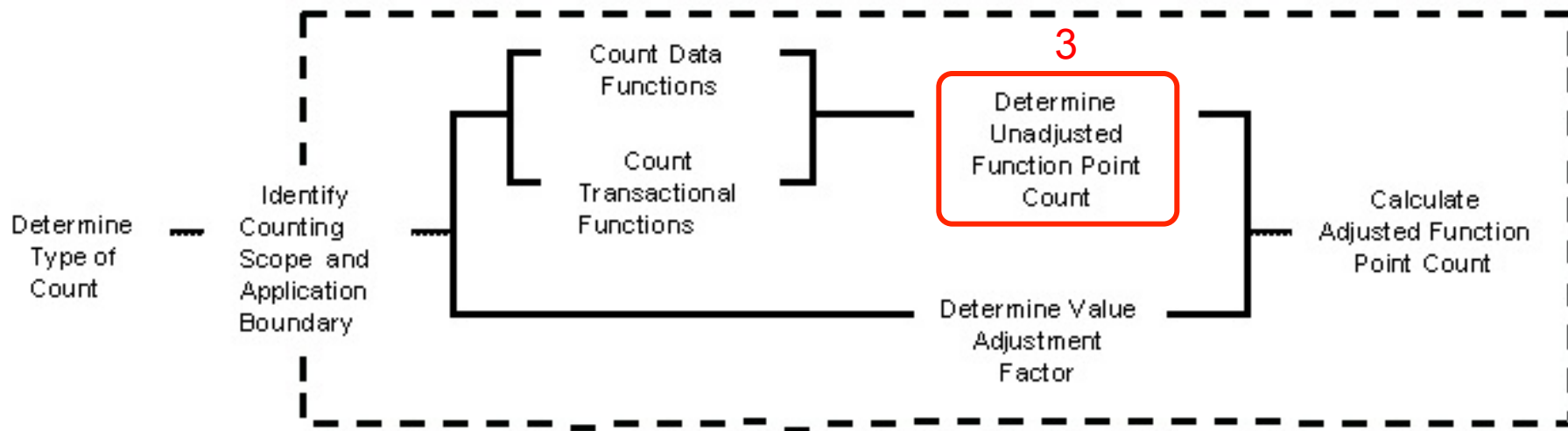


Procedure Diagram

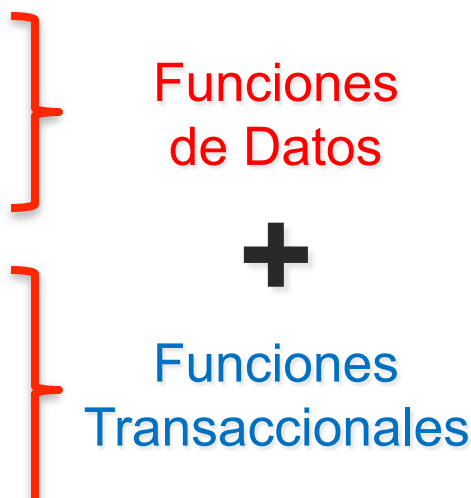


1. Determinar el tipo de conteo de PFs
 - *Desarrollo del proyecto* → Asociado con la construcción de un nuevo sistema
 - *Mantenimiento del proyecto* → Asociado con la modificación del proyecto luego de haber sido puesto en producción
 - *Aplicación* → Asociado con el cálculo de tamaño de aplicaciones en producción
2. Identificar el alcance y la frontera de la aplicación

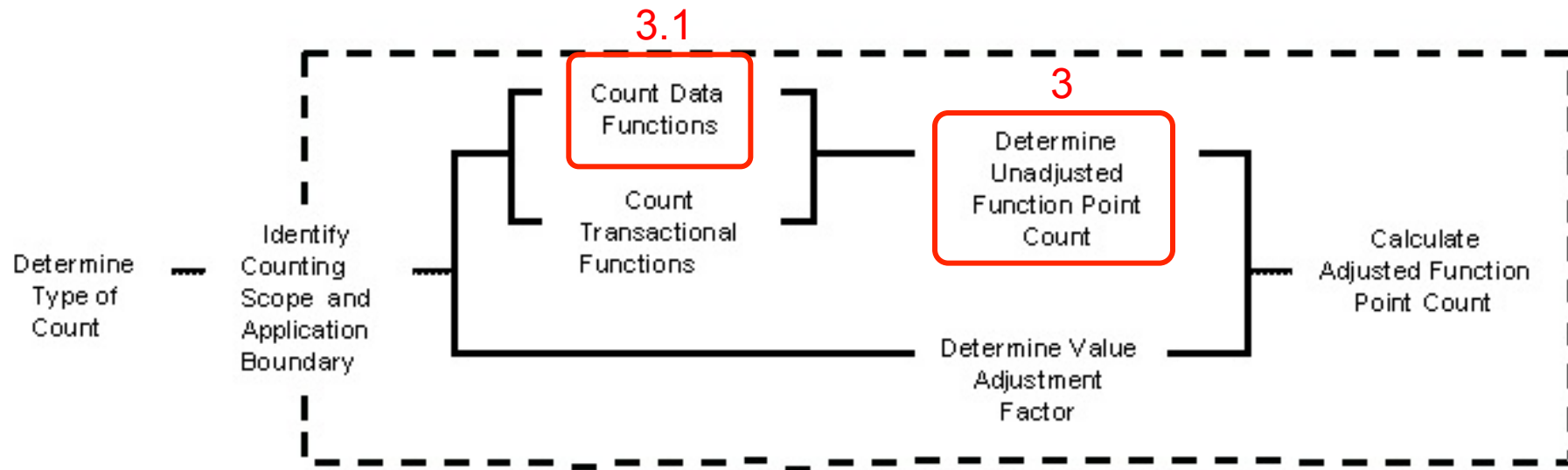
Procedure Diagram



3. Determinar los *puntos de función sin ajuste*

- Los PSA corresponden a una medida de la *funcionalidad* de la aplicación desde el punto de vista de un usuario
 - Sub-categorías:
 - Archivos Lógicos Internos (ILF)
 - Archivos de Interfaz Externos (EIF)
 - Entradas Externas (EI)
 - Salidas Externas (EO)
 - Consultas Externas (EQ)
- 

Procedure Diagram



- Calcular **Funciones de Datos**
 - Representan la funcionalidad que satisfacen requerimientos de datos internos y externos
 - Pasos:
 - Identificar archivos
 - Asignar a cada uno un tipo (ILF, EIF)
 - Identificar la cantidad de RET y DET
 - Asignar a cada uno un valor de complejidad (Alta, Media, Baja) en función de la cantidad de RET y DET

- Calcular **Funciones de Datos**

Definiciones

- *Archivo Lógico Interno (ILF)* → Grupo de datos, lógicamente relacionado, identificable por el usuario y mantenido dentro de la frontera de la aplicación
- *External Interface File (EIF)* → Grupo de datos, lógicamente relacionado, identificable por el usuario, referenciado por la aplicación, pero mantenido fuera de la frontera de la aplicación.
- *Nota:* Un EIF para una aplicación tiene que ser un ILF para alguna otra

- Calcular **Funciones de Datos**

Data Element Type (DET)

- Es un campo único (no repetitivo) reconocible por el usuario
- Contar un DET por cada campo no repetitivo, reconocible por el usuario, que se recupera o mantiene desde ILF o EIF a través de un proceso
- Ejemplo:

Para un archivo “Empleado” identificado (sea ILF o EIF), los DETs que pueden ser requeridos generalmente son:

Nombre del empleado, Fecha-de-Nacimiento, Genero, Cargo, Jefe, Salario

- Calcular **Funciones de Datos**

Record Element Type (RET)

- Es un subconjunto de campos de un archivo, reconocible como tal por el usuario
- Ejemplo:

Aplicación de RRHH → Empleado tiene datos generales y además puede ser mensual o jornalero. Adicionalmente, puede tener personas a su cargo (núcleo familiar)

RETs:

- Mensual (incluyendo datos generales) → *Obligatorio*
- Jornalero (incluyendo datos generales) → *Obligatorio*
- Núcleo Familiar → *Opcional*

- Calcular **Funciones de Datos**

Caracterización de la complejidad

Para ILF/EIF	1 a 19 DET	20 a 50 DET	51 o más DET
1 RET	Baja	Baja	Media
2 a 5 RET	Baja	Media	Alta
6 o más RET	Media	Alta	Alta

Contribución de datos

Complejidad / Tipo de Archivo	Baja	Media	Alta
Archivo Lógico Interno (ILF)	7	10	15
Archivo de Interfaz Externa (EIF)	5	7	10

- Calcular **Funciones de Datos**

Ejemplo:

Una aplicación que mantiene los siguientes archivos:

Tarea (#tarea, nom_tarea, escala)

Descripcion_Tarea (#tarea, #categoria, descripcion)

Empleado (id, nom_empleado, fecha_nac, fecha_ingreso, #tarea)

ILF identificados: Tarea, Empleado

Tarea:

2 RET → Tarea, Descripcion_Tarea

5 DET → #tarea, nom_tarea, escala, #categoria, descripcion

Empleado:

1 RET

5 DET → id, nom_empleado, fecha_nac, fecha_ingreso, #tarea

- Calcular **Funciones de Datos**

Ejemplo:

Contribución de Datos

Archivo	Tipo	Nivel Complejidad	Cuenta
Empleado	ILF	Baja	7
Tarea	ILF	Baja	7
Total de Contribución de Datos:			14

- Calcular **Funciones de Datos**
- Contribución de Datos – Guía
 - ¿Los datos son un grupo lógico que soporta requerimientos del usuario?
- Una aplicación puede usar un mismo ILF o EIF en múltiples procesos, pero el archivo se cuenta una sola vez
- Un mismo archivo no se puede contar a la vez como ILF y EIF; si cumple ambos criterios, contarlo como ILF
- Si un grupo de datos no fue contado como ILF ni EIF, contar sus DET para el ILF o EIF que incluye al grupo
- No asumir que un archivo físico, tabla o clase de objetos corresponde a un archivo lógico desde el punto de vista del usuario
- No asumir que todo archivo físico debe ser contado o incluido como parte de un ILF o EIF

- Calcular **Funciones de Datos**
- Contribución de Datos – Guía
 - ¿Dónde se mantienen los datos, dentro o fuera de la aplicación?
- Archivos lógicos mantenidos por más de una aplicación se consideran como ILF al contar cada una
- Recordar que en el caso anterior, en cada aplicación sólo se consideran los DET que usa y estos se determinan desde el punto de vista de cada aplicación

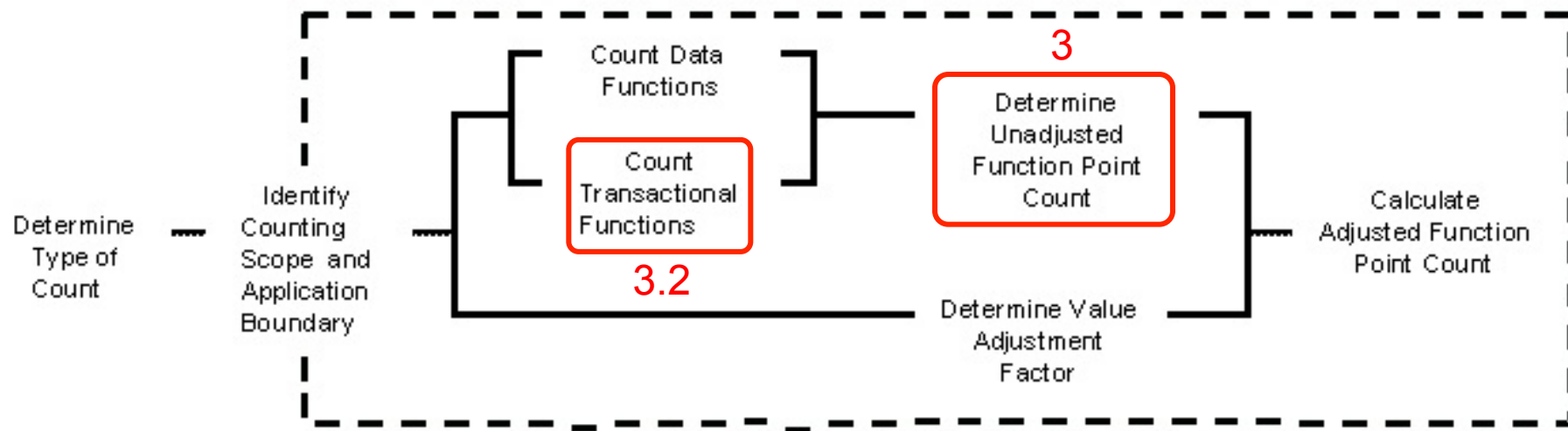
- Calcular **Funciones Transaccionales**
 - Representan la funcionalidad proporcionada al usuario para procesar datos
 - Pasos:
 - Identificar transacciones
 - Asignar a cada una un tipo (EI, EO, EQ)
 - Identificar la cantidad de DET y FTR
 - Asignar a cada una un valor de complejidad (Alta, Media, Baja) en función de la cantidad de DET y FTR

- Calcular **Funciones Transaccionales**

File Type Referenced (FTR)

- Es un tipo de archivo al que se hace referencia en una transacción; tiene que ser un ILF o EIF

Procedure Diagram



- Calcular **Funciones Transaccionales**

Tipos de Transacciones

- *EI (External Input) - Entrada Externa* → Proceso elemental en el que datos cruzan la frontera de la aplicación de afuera hacia adentro. La intención primordial es mantener uno o más ILF y/o alterar el comportamiento del sistema
- *EO (External Output) - Salida Externa* → Proceso elemental en el que datos derivados a partir de uno o más ILF o EIF cruzan la frontera de adentro hacia fuera. Un EO **puede** actualizar un ILF o alterar el comportamiento del sistema
- *EQ (External Query) - Consulta Externa* → Proceso elemental en el que datos o información de control cruzan la frontera de adentro hacia fuera. NO incluye datos derivados y NO mantiene ningún ILF y NO altera el comportamiento del sistema

- Calcular **Funciones Transaccionales**

Tipos de Transacciones - Resumen

Función	EI	EO	EQ
Altera el comportamiento del sistema	IP	O	NO
Mantiene uno o más ILF	IP	O	NO
Presenta información al usuario	O	IP	IP
Presenta datos derivados al usuario	O	IP	NO

IP = Intención Primordial

O = Opcional

- Calcular **Funciones Transaccionales**

Transacciones – Unicidad

Se cuenta si se cumple al menos uno de los siguientes:

- **Para EI:**
 - Lógica distinta de otras EI
 - El conjunto de DET distinto del de otras EI
 - Conjunto de ILF o EIF distinto del de otras EI
- **Para EO, EQ:**
 - Lógica distinta de otras EO o EQ
 - El conjunto de DET distinto del de otras EO o EQ
 - Conjunto de ILF o EIF distinto del de otras EO o EQ

- Calcular **Funciones Transaccionales**

Complejidad de Transacciones – FTR

- Contar un FTR por cada ILF mantenido
- Contar un FTR por cada ILF o EIF leído durante el proceso del EI
- Contar sólo un FTR por cada ILF que es leído y mantenido

Ejemplo: Retiro de una cuenta bancaria

ILF en la aplicación:

- Cuenta
- Movimientos
- Cotizaciones dólar

2 FTR

El proceso de retiro lee la cuenta, verifica saldo, graba movimiento y actualiza la cuenta

- Calcular **Funciones Transaccionales**

Complejidad de Transacciones – DET

- Contar un **DET** por cada campo reconocible por el usuario, no repetido, que entra o sale de la aplicación atravesando su frontera y es requerido para completar el EI
- No contar campos leídos o derivados por la aplicación y almacenados en un **ILF** si los campos no cruzaron la frontera
- Contar un **DET** por la posibilidad de que el sistema envíe un mensaje fuera de la frontera de la aplicación para indicar un error, confirmar que el proceso está completo o verificar si el proceso debiera continuar
- Contar un **DET** por la posibilidad de especificar una acción, lo mismo si hay múltiples métodos para invocar el mismo proceso lógico

- Calcular **Funciones Transaccionales**

Complejidad de Transacciones – DET

Ejemplo 1 → Agregar un empleado con los datos:

- Nombre
- Fecha de ingreso
- Identificación
- Fecha de nacimiento

4 DET

Ejemplo 2 → Ingreso de datos de factura de proveedor:

- Código proveedor (E)
- Nombre proveedor (S)
- Fecha factura (E)
- Importe total (E)
 - Código artículo, precio unitario, cantidad

7 DET

- Calcular **Funciones Transaccionales**

Caracterización de la complejidad

Para EI	1 a 4 DET	5 a 15 DET	16 o más DET
0 a 1 FTR	Baja	Baja	Media
2 FTRs	Baja	Media	Alta
3 o más FTRs	Media	Alta	Alta

Para EO / EQ	1 a 4 DET	5 a 15 DET	16 o más DET
0 a 1 FTR	Baja	Baja	Media
2 a 3 FTRs	Baja	Media	Alta
4 o más FTRs	Media	Alta	Alta

- Calcular Funciones Transaccionales

Contribución de datos

Complejidad / Tipo de Transacción	Baja	Media	Alta
External Input (EI)	3	4	6
External Output (EO)	4	5	7
External inQuiry (EQ)	3	4	6

- Calcular **Funciones Transaccionales**

Ejemplo:

Una aplicación integrada por:

Alta cliente (#cliente, nombre, dirección)

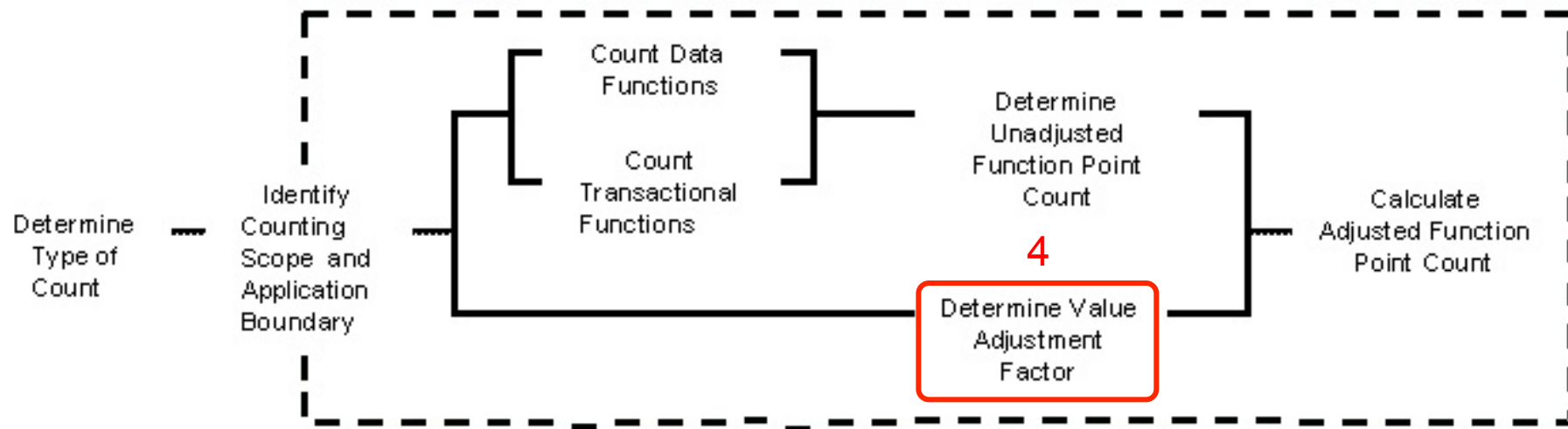
Listado de clientes (#cliente, nombre, dirección)

Consulta de la cantidad de clientes existentes

ILF identificados: Cliente

Transacción	Tipo	Nivel Complejidad	Cuenta
Alta Cliente	EI	Baja	3
Listado Clientes	EQ	Baja	3
Cantidad Clientes	EO	Baja	4
Total de Contribución de Transacciones:			10

Procedure Diagram



4. Determinar el *factor de complejidad técnica* para ajuste

- Medida del grado de influencia en la complejidad del sistema, de una serie de factores preestablecidos
- Se consideran **14** factores diferentes
- Cada característica tiene asociadas descripciones que ayudan a determinar el grado de influencia de dicha característica
- Cada factor se evalúa en una escala de 0 (no tiene influencia) a 5 (influencia muy fuerte)

Factor de Complejidad Técnica (FCT)

- Permite ajustar el total de puntos de función en **35%** (hacia arriba o hacia abajo) de acuerdo con la complejidad de procesamiento
- Se calcula con la fórmula:

$$\text{Factor de Complejidad Técnica} = 0.65 + (0.01 \times \text{Puntos de Complejidad Técnica})$$

- Su rango de valores está entre **0.65** (0 puntos de complejidad) hasta **1.35** (70 puntos de complejidad)

Puntos de Complejidad Técnica (FCT)

- Se estima el grado de influencia que cada uno de los 14 factores de complejidad de procesamiento tiene en la implantación del sistema, y se suman estos valores
- Valores posibles para estimar el grado de influencia:
 - 0 = No está presente, no tiene ninguna influencia si lo está
 - 1 = Influencia poco significativa
 - 2 = Influencia moderada
 - 3 = Influencia medianamente significativa
 - 4 = Influencia significativa
 - 5 = Influencia muy fuerte, en toda la extensión del sistema

Factores de Complejidad Técnica (FCT)

1. Comunicación de datos
2. Actualización en línea
3. Servicios distribuidos
4. Procesamiento complejo
5. Desempeño
6. Reusabilidad
7. Ambiente de uso sobrecargado
8. Facilidad de instalación
9. Rata de transacciones
10. Facilidad de operación
11. Entrada de datos en línea
12. Múltiples lugares de operación
13. Eficiencia del usuario final
14. Facilidad de modificación

Factores de Complejidad Técnica (FCT)

1. **Comunicación de datos** → Los datos y la información de control utilizados por la aplicación son enviados o recibidos por medio de servicios de comunicación electrónica
2. **Actualización en línea** → La aplicación ofrece actualización en línea de los almacenamientos de datos lógicos
3. **Servicios distribuidos** → La aplicación mantiene datos distribuidos (en múltiples computadores) o procesa información en forma distribuida

Factores de Complejidad Técnica (FCT)

4. **Procesamiento complejo** → Muchas interacciones de control y puntos de decisión, uso extensivo de operaciones lógicas y matemáticas, amplia necesidad de código de manejo de casos excepcionales resultante de operaciones incompletas o fallidas
5. **Desempeño** → Los requisitos de desempeño de la aplicación, ya sea en tiempo de respuesta o capacidad de procesamiento de datos, tienen influencia en su diseño, desarrollo, instalación o soporte

Factores de Complejidad Técnica (FCT)

6. Reusabilidad → El código de la aplicación será diseñado específicamente para ser reutilizado en otras aplicaciones
7. Ambiente de uso sobrecargado → Se desea correr la aplicación en equipo existente o comprometido que tendrá una alta carga de operación
8. Facilidad de instalación → La facilidad de conversión o instalación, aparece como parte de los requisitos
9. Rata de transacciones → La rata de transacciones es alta y tendrá influencia en el diseño, desarrollo, instalación o soporte de la aplicación

Factores de Complejidad Técnica (FCT)

- 10. **Facilidad de operación** → Se ofrecerán procedimientos efectivos de arrancada, copias de respaldo (back-up) y recuperación y éstos serán probados en la fase de pruebas. La aplicación debe reducir al mínimo la necesidad de intervención del operador
- 11. **Entrada de datos en línea** → La aplicación ofrece servicios de entrada de datos y control en línea
- 12. **Múltiples lugares de operación** → La aplicación será diseñada específicamente para ser instalada en varios lugares de operación y/o en varias organizaciones

Factores de Complejidad Técnica (FCT)

- 13. Eficiencia del usuario final → Las operaciones en línea ofrecidas, deben diseñarse o implantarse con un énfasis en la eficiencia de los usuarios
- 14. Facilidad de modificación → La aplicación será diseñada específicamente para ser flexible a los cambios (i.e. altamente parametrizada, fácilmente modificable, etc)

Factor de Complejidad Técnica (FCT)

Ejemplo de cálculo:

- | | |
|--|---|
| 1. Comunicación de datos - 5 | 8. Facilidad de instalación - 1 |
| 2. Actualización en línea - 5 | 9. Rata de transacciones - 5 |
| 3. Servicios distribuidos - 5 | 10. Facilidad de operación - 3 |
| 4. Procesamiento complejo - 3 | 11. Entrada de datos en línea - 3 |
| 5. Desempeño - 4 | 12. Múltiples lugares de operación - 4 |
| 6. Reusabilidad - 0 | 13. Eficiencia del usuario final - 5 |
| 7. Ambiente de uso sobrecargado - 1 | 14. Facilidad de modificación - 3 |

Factor de Complejidad Técnica (FCT)

Ejemplo de cálculo:

- Total de Puntos de Complejidad Técnica (PCT) = 47

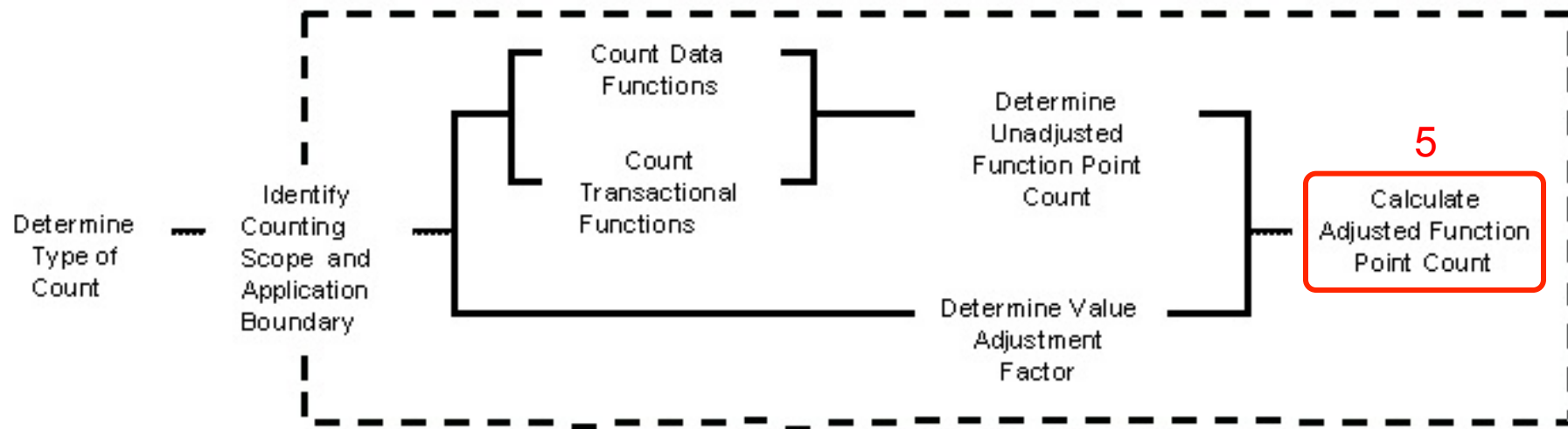
$$\text{Factor de Complejidad Técnica} = 0.65 + (0.01 \times \text{Puntos de Complejidad Técnica})$$

$$FCT = 0.65 + (0.01 \times 47)$$

$$FCT = 0.65 + 0.47$$

$$\mathbf{FCT = 1.12}$$

Procedure Diagram



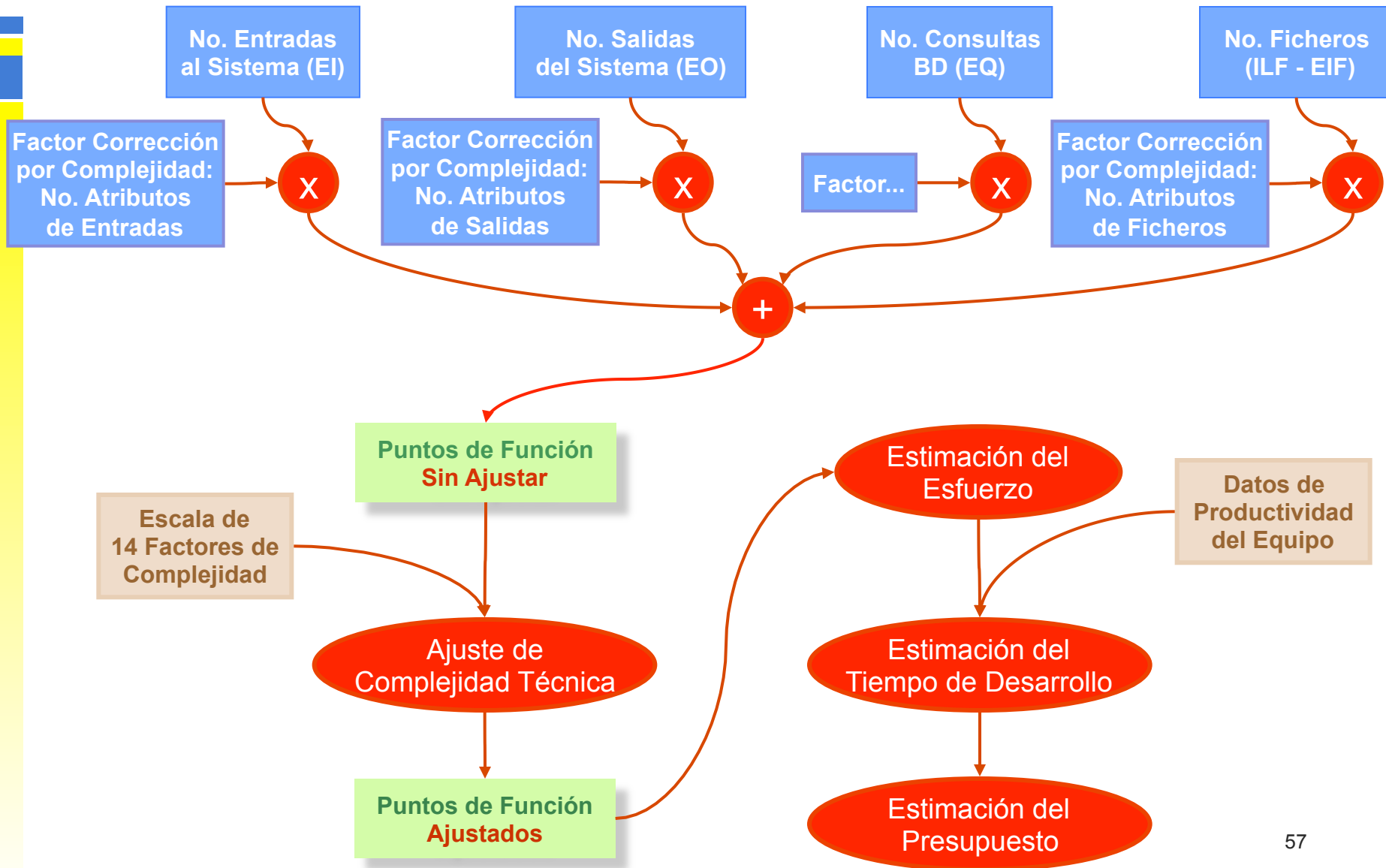
5. Calcular los *puntos de función ajustados*

- Son calculados por medio de una fórmula de acuerdo con el tipo de conteo seleccionado
 - Desarrollo del proyecto
 - Mantenimiento del proyecto
 - Aplicación
- Para desarrollo de un nuevo proyecto la fórmula es:

$$PF = PSA * FCT$$

- Introducción
- Historia
- Puntos de Función Vs. LOCs
- Cálculo de Puntos de Función (PF)
- **Proceso de estimación mediante PF**
- Referencias

Proceso de Estimación con PFs



Calculo del Esfuerzo

- La productividad de un equipo de desarrollo está dada por:

$$\text{Productividad del equipo} = \frac{\text{Tamaño de la aplicación}}{\text{Esfuerzo total}}$$

- Despejando de allí el esfuerzo total tenemos:

$$\text{Esfuerzo total} = \text{Tamaño de la aplicación} * \text{Productividad del equipo}$$

Calculo del Esfuerzo

- El tamaño de la aplicación está determinada por el resultado del proceso de estimación por PF explicado anteriormente
- Para calcular el esfuerzo faltaría conocer la productividad del equipo:
 - Mediante datos históricos de la organización de proyectos anteriores cuya estimación haya sido también PF
 - Mediante la normalización de datos históricos de LOCs de proyectos anteriores de la organización
 - Mediante datos históricos promedio de productividad sobre la tecnología que estamos trabajando.
Por ejemplo: Tablas de lenguajes de Capers Jones
<http://www.spr.com>

Calculo del Esfuerzo – Ejemplo

Entorno y Lenguaje	Esfuerzo	
	Líneas de Código por PF	Horas por PF
Lenguajes 2GL: Ensamblador, C,...	300	20 a 30
Lenguajes 3GL: Cobol	100	10 a 20
Lenguajes 4GL: VisualXX	20	5 a 10

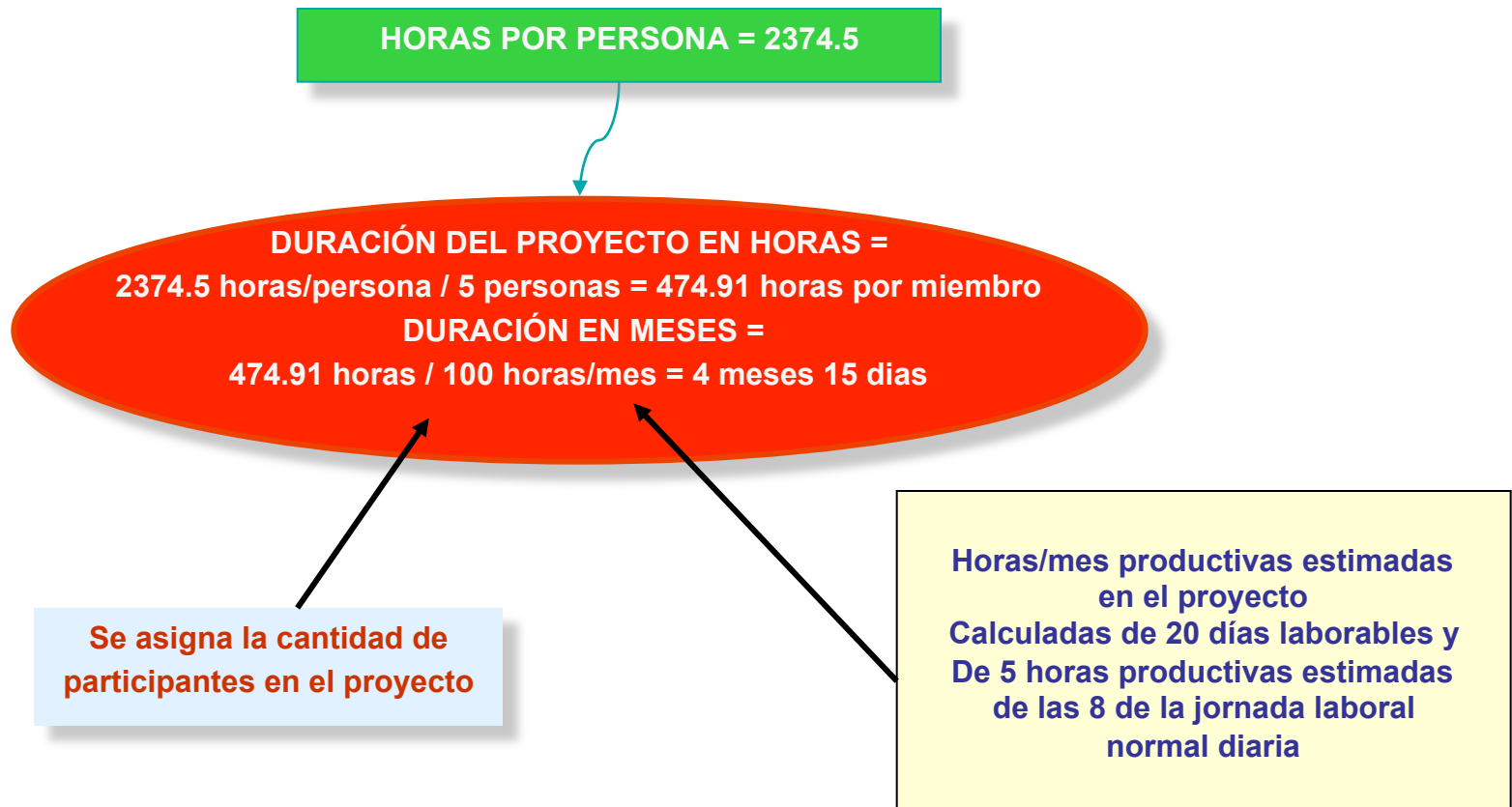
PFA = 296.82

Cambiar horas/
efectivas por
horas productivas
estimadas

LOCs = PFA * (LOCs POR PF)

**Esfuerzo horas/persona =
 $PFA / [1 / 8 \text{ persona / hora}] = 296.82 / 0.125$
= 2374.5 horas/persona**

Calculo de duración del proyecto – Ejemplo



Calculo de presupuesto del proyecto – Ejemplo

DURACIÓN DEL PROYECTO EN MESES = 5 meses

Participante 1: Sueldo

Participante 2: Sueldo

Participante n: Sueldo

Costo Total del Proyecto =

**sueldos 1 participante del proyecto * 5 participantes * 5 meses
+
Otros costos necesarios durante la realización del proyecto
= 2000 * 5 * 5 = 50000**

**En la práctica
se deben especificar
Otros costos de operación
para determinar el presupuesto
total del proyecto**

- El tamaño de la aplicación puede estimarse correlacionando los PF con el número de líneas de código, aunque es recomendable efectuar la estimación sin necesidad de normalizar la métrica
- Se puede usar la *técnica de regresión lineal* con base en información de proyectos anteriores. Este método puede resultar mucho más confiable
- Los PF pueden contarse antes y después del desarrollo, con lo cual es posible hacer correlaciones que permitan ajustar las estimaciones para lograr una mayor precisión

- La correlación entre los PF y el tamaño de las aplicaciones es normalmente alta
- Los puntos de función son un método relativamente sencillo para estimar el tamaño de una aplicación
- Los puntos de función son un proxy de estimación muy conveniente, ya que son fácilmente observables a lo largo del desarrollo

1. *Software Sizing, Estimation, and Risk Management: When Performance is Measured Performance Improves*. Daniel D. Galorath, Michael W. Evans. Auerbach Publications. 1 edition. March 15, 2006
2. *Practical Software Estimation: Function Point Methods for Insourced and Outsourced Projects*. M. A. Parthasarathy. Addison-Wesley Professional. 1 edition. March 8, 2007
3. *Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort & Duration*. Peter Hill, International Software Benchmarking Standards Group. McGraw-Hill Osborne Media. 1 edition. September 10, 2010
4. International Function Point Users Group (IFPUG) Home Page, <http://www.ifpug.org/home/docs/ifpughome.html>
5. IFPUG, Guidelines to Software Measurement, IFPUG, 1999