UNIDAD 4

Ingeniería de Software

Estimaciones de Software

v. 4.0

Conceptos de Estimaciones de Software

Fundamentos de Estimaciones

¿ Qué preguntas nos hacemos al estimar ?

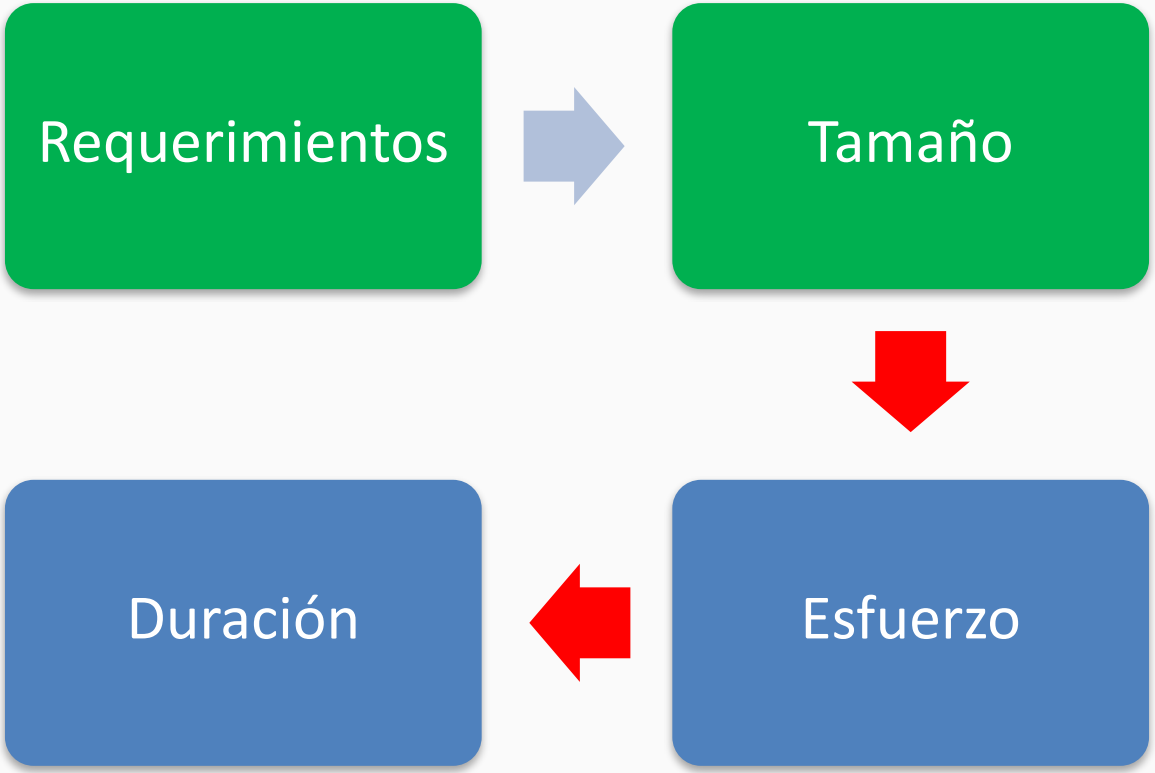
- *La mayoría de las veces nos preguntamos:*
  - ¿ Cuánto ***tiempo*** vamos a tardar en desarrollar el proyecto ?
  - ¿ Cuantas ***hs/pers*** se necesitan para desarrollarlo ?
  - ¿ Cuánto ***va a costa***r el proyecto ?

Son pocas las veces que nos preguntamos:

- *¿ Cuál es el* **tamaño** *de lo que tenemos que construir ?*
  - Esta pregunta debería ser la primera para poder pasar a las anteriores !!!
  - La realidad parece indicarnos que el tamaño se calcula informalmente y solo formalizamos el esfuerzo, ventana y costo
  - Debemos ser lo mas rigurosos posibles con la estimación del tamaño

Fundamentos de Estimaciones

EL PROCESO DE ESTIMAR



Fundamentos de Estimaciones

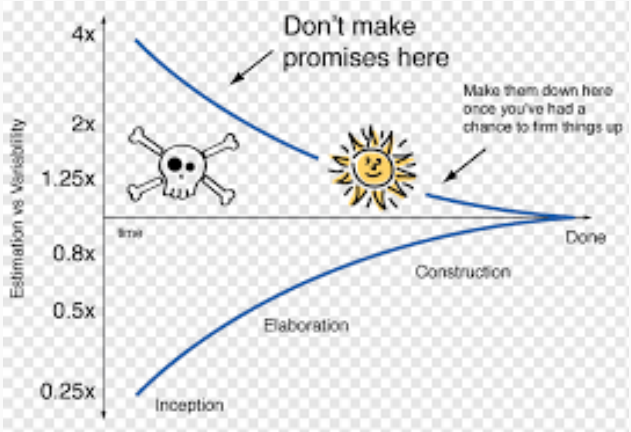
¿Por qué fallan las estimaciones?

- *Optimismo*
- *Estimaciones informales (“estomacales”)*
- *No hay historia*
- *Mala definición del alcance*
- *Novedad / Falta de Experiencia*
- *Complejidad del problema a resolver*
- *No se estima el tamaño*
- *Porque la estimación fue buena pero cuando empieza el proyecto:*
    - *Mala administración de los requerimientos*
    - *No hay seguimiento y control*
    - *Se confunde progreso con esfuerzo*

Fundamentos de Estimaciones

“Nivel de incertidumbre”

- *El “cono de la incertidumbre” define niveles estadísticos predecibles de la incertidumbre de las estimaciones en cada etapa del proyecto*
- *Cuanto más refinada la definición, mas exacta será la estimación*



Fundamentos de Estimaciones

Tips para estimar …

- *Diferenciar entre estimaciones, objetivos y compromisos*
- *Asociar a las estimaciones un % de confiabilidad*
- *Es recomendable presentar las estimaciones como rangos en lugar de un único valor*
- *Siempre presentar junto con la estimación, los supuestos que se tuvieron en cuenta para llegar a la misma*

Fundamentos de Estimaciones

Tips para estimar … (cont.)

- *Tener presente la* **Ley de Parkinson***: "Toda tarea se expande hasta ocupar el tiempo que tiene asignado"*
- *Considerar todas las actividades relacionadas al desarrollo de sw, no solamente codificación y testing (análisis, diseño, actividades de SCM, testing, etc.)*
- *No asumir que solo por el paso del tiempo y de las fases de un proyecto se avanza con menor incertidumbre en las estimaciones (Cono de la incertidumbre)*
- *Recolectar datos históricos para tener como referencia*

Fundamentos de Estimaciones

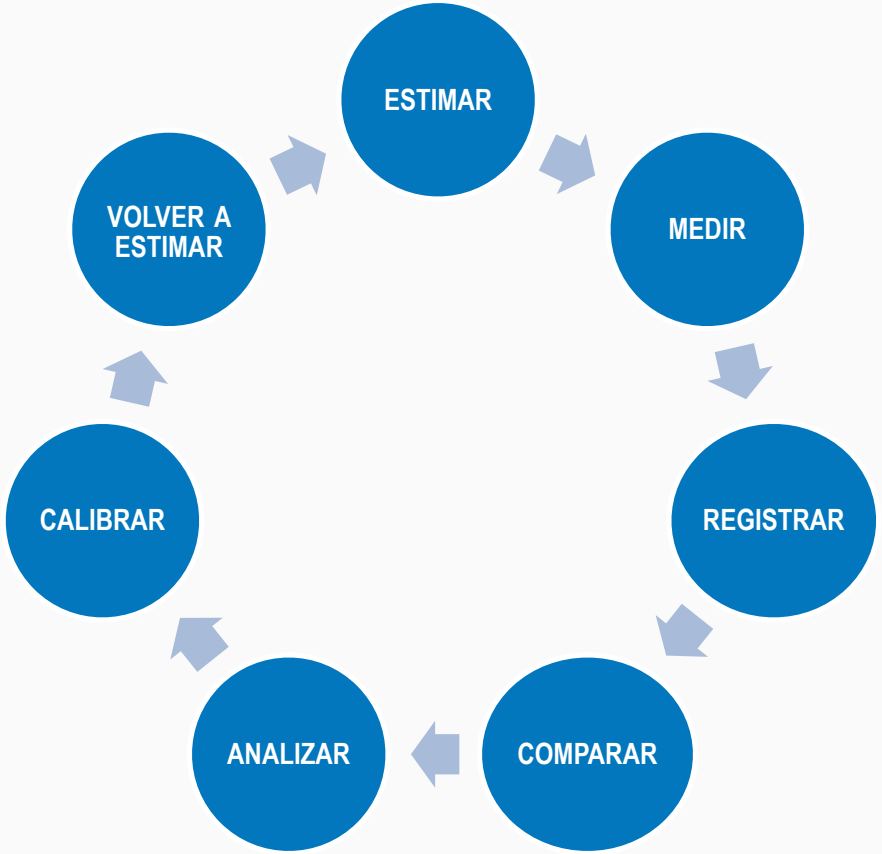
Estimaciones creíbles …

- *Las estimaciones las hacen las personas, no herramientas ni modelos.*
  - Se necesitan juicios razonables y compromisos con los objetivos organizacionales que no se pueden delegar a modelos automáticos
- *Las estimaciones se basan en comparaciones.*
  - Cuando las personas estiman buscan similitudes y diferencias con proyectos previos
- *Para que la gente pueda estimar necesitamos* **historia** *de proyectos pasados para poder comparar*
  - Las estimaciones se pueden mejorar colectivamente juntando historia
- *Un método creíble debe ser de caja blanca*

Fundamentos de Estimaciones

Ciclo de Estimación

- ***Estimar**: Comenzar x estimar el tamaño para derivar el esfuerzo y el costo*
- ***Medir**: Mientras evoluciona el proyecto, medir el tamaño, el esfuerzo y costo incurrido*
- ***Registrar**: Dejar claras las mediciones tomadas*
- ***Analizar**: Razones de desvíos, supuestos que quizás variaron, temas no contemplados, etc...*
- ***Calibrar:** Ajustar c/u de las variables y parámetros que intervienen en el proceso de estimación*
- ***Volver a estimar:***
    - *El mismo proyecto pero ahora con mas información que al comienzo del mismo*
    - *Nuevos proyectos con el proceso ajustado por la "calibración"*



Métodos de Estimación

Métodos de Estimación

- **Métodos no paramétricos**
  - *Juicio Experto*
  - *Pert*
  - *Wideband Delphi*
  - *Planning Poker*

- **Métodos parámetricos**
  - *Function Points*
  - *Use Case Points*
  - *Objects Ponts*

Métodos de estimación

Recomendaciones - ¿Qué hacer?

- *Juntar historia de todos los proyectos*
  - *Registrar todas las lecciones aprendidas*
  - *Tener en cuanta la importancia de la comunicación*
- *Desarrollar un método de estimación adecuado a la instalación*
  - *Basarse en los conocidos y crear el propio*
  - *Lo importante es su eficacia*
- *Apoyarnos en herramientas*
  - *Permiten ayudarnos en la implementación del “ciclo dorado” para implementar un método*

Métodos de estimación

Recomendaciones - ¿Qué NO hacer?

- *Descartar los métodos (Rechazarlos porque los consideramos inaplicables*
  - *Rechazarlos porque los consideramos inaplicables*
- *Utilizar métodos elaborados sin experiencia o información suficiente*
  - *Los métodos elaborados requieren de habilidades que hay que desarrollar*
- *Ignorar los supuestos*
  - *Siempre deben estar claro al comienzo, dejando claro el nivel de certidumbre de la estimación.*

Preguntas

Apéndices

Métodos no paramétricos

- Juicio Experto
  - Depende de la persona que haga la estimación
  - Se basa en la experiencia personal
  - Por lo general se recurre a analogías

- Método Pert (Program Evaluation and Review Technique)
  - Se basa en el juicio experto
  - Incluye los factores optimistas y pesimistas en su estimación
  - Estimación = (Optimista + 4 x Medio + Pesimista) / 6
  - Se puede usar para estimar tamaño y esfuerzo
  - También se conoce como el método Clark

Métodos no paramétricos

- Wideband Delphi
  - Es el juicio experto en grupos
  - Da lugar a la participación de todos los involucrados
  - Ventajas: fácil de implementar y da sentido de propiedad sobre la estimación
  - Se puede utilizar en etapas tempranas del proyecto
  - Se recomienda utilizarlo en proyectos poco conocidos en donde no se cuenta con historia

Métodos Paramétricos

OBJECT POINTS

- *Los “objetos” contemplan pantallas, reportes & módulos*
  - No está relacionado necesariamente con objetos de OOP
  - *El método original contempla solo estos tres tipos de objetos*
    - Implementaciones ad-hoc del método consideran otros tipos de componentes como (stored procedures, clases, scripts SQL, etc …)
- *Se asigna a cada componente un peso de acuerdo a su clasificación por complejidad*
- *Considera como factor de ajuste el porcentaje de reuso de código*

Métodos Paramétricos

OBJECT POINTS (cont.)

- *Cada objeto es clasificado de acuerdo a su nivel de complejidad en:*
  - Simple
  - Medio
  - Difícil

	Number and sources of data tables		
Number of Views Contained	**Total < 4**	**Total < 8**	**Total 8+**
<3	simple	simple	medium
3-7	simple	medium	difficult
8+	medium	difficult	difficult

	Number and source of data tables		
Number of Sections Contained	**Total < 4**	**Total < 8**	**Total 8+**
0-1	simple	simple	medium
2-3	simple	medium	difficult
4+	medium	difficult	difficult

Métodos Paramétricos

OBJECT POINTS (cont.)

- *Luego se le brinda un peso a cada nivel de complejidad (teniendo directa proporción con el esfuerzo que requiere la implementación de c/u)*

	Weight		
Type	**Simple**	**Medium**	**Difficult**
Screen	1	2	3
Report	2	5	7
Modules	10	10	10

- *Finalmente sumando la cantidad de objetos de cada complejidad con sus respectivos pesos nos da como resultado los **OPs***

- *Teniendo en cuenta el reuso → **NOP (NewOP)** = OP * [(100 - % Reuso) / 100]*

Métodos Paramétricos - Function Points

- Miden el tamaño del SW en base a la funcionalidad capturada en los requerimientos
- Usa el modelo lógico o conceptual para trabajar
- En general son aceptados en la industria a pesar de su complejidad y antigüedad
- Hay muchas heurísticas en el mercado basadas en PFs
- Los PFs son independientes de la tecnología
- Requieren un análisis de requerimientos avanzado

Métodos Paramétricos

- Elementos del sistema
  - Archivos lógicos internos (ALI o ILF)
    - Grupo de datos lógicos de información o de control, identificables por el usuario, mantenidos a través de procesos elementales de la aplicación dentro de los límites de la misma.
  - Archivo de interface externa (AIE o EIF)
    - Grupo de datos lógicos de información o de control, identificables por el usuario, referenciados por la aplicación pero mantenidos a través de procesos elementales de una aplicación diferente.
    - Diferencia: Los EIF no son mantenidos por la aplicación.
  - Entradas externas (EE o EI)
    - Proceso elemental de la aplicación que procesa datos o información de control que entran desde el exterior del sistema.
    - Los datos procesados mantienen uno o más ILFs.
    - La información de control puede o no ser mantenida en un ILF.

Métodos Paramétricos - Function Points

- Elementos del sistema (Continuación)
  - Salidas externas (SE o EO)
    - Es un proceso elemental de la aplicación que envía datos o información de control fuera de los límites del sistema.
    - Puede hacer update de un ILF.
    - Dicho procesamiento de información debe contener al menos una fórmula matemática o cálculo, o crear datos derivados.
    - Una EO puede también mantener uno o más ILFs y/o alterar el comportamiento del sistema.
  - Consultas Externas (CE o EQ)
    - Es un proceso elemental de la aplicación que envía datos o información de control fuera de los límites del sistema.
    - El proceso elemental NO posee fórmulas matemáticas ni cálculos. Y no crea datos derivados.
    - Una EQ no mantiene ILFs durante su procesamiento, ni altera el comportamiento del sistema.

Métodos Paramétricos - Function Points

- Unadjusted function points (UFP), se calculan de la siguiente manera:

Function Type	Low	Average	High
External Input	x3	x4	x6
External Input	x4	x5	x7
Logical Internal File	x7	x10	x15
External Interface File	x5	x7	x10
External Inquiry	x3	x4	x6

	1-5 Data element types	6-19 Data element types	20+ Data element types
0-1 File types referenced	Low	Low	Average
2-3 File types referenced	Low	Average	High
4+ File types referenced	Average	High	High

Métodos Paramétricos - Function Points

- Calcular el technical complexity factor (TCF):
  TCF = 0.65 + (sum de factores) / 100

  Hay 14 factores de complejidad técnica. Cada uno se evalua de acuerdo al grado de influencia (0-5).
  - Data communications
  - Performance
  - Heavily used configuration
  - Transaction rate
  - Online data entry
  - End user efficiency
  - Online update
  - Complex processing
  - Reusability
  - Installation ease
  - Operations ease
  - Multiple sites
  - Facilitate change
  - Distributed functions

- El TCF también se conoce como el CGS, Características Generales del Sistema

Métodos Paramétricos - Function Points

- **Cálculo de puntos de función**
  - *Puntos de función netos = PFN = UFP* * TCF

- Conversión de FP a LOC
  - En base a estandares del Mercado, 1 FP equivale a:

	Mínimo	Media	Máximo
Java	40 LOC	55 LOC	80 LOC
C++	40 LOC	55 LOC	80 LOC
Cobol	65 LOC	107 LOC	150 LOC
SQL	7 LOC	13 LOC	15 LOC

Métodos Paramétricos - Use Case Points

- El número de Use Case Points depende de :
  - Cantidad y complejidad de los casos de uso
  - Cantidad y complejidad de los actores intervinientes en el sistema
  - Factores técnicos y ambientales

- El método requiere que sea posible contar el número de transacciones en cada caso de uso. Una transacción es un evento que ocurre entre el actor y el sistema.

- Basado en el método de Function Points

- Elementos del sistema
  - Casos de Uso
  - Transacciones de Casos de Uso
  - Actores

Métodos Paramétricos - Use Case Points

- **Actores Simples:**
  - Son sistemas externos, son muy predecibles, todo sistema con interfaz de aplicación bien definida.
- **Actores Promedio:**
  - Dispositivos de Hardware. Requieren más esfuerzo controlarlos, son más propensos a errores.
  - Personas interactuando a través de una interfaz basada en texto
- **Actores Complejos:**
  - Son humanos, son impredecibles y difíciles de controlar. Personas que interactúan a
  - través de una GUI
- Se cuenta el número de actores en cada categoría, multiplicando ese número por el correspondiente peso y se obtiene el UAW (Unadjusted Actor Weigth).

Tipo de Actor	Peso
Simple	1
Medio	2
Complejo	3

Métodos Paramétricos

- Los casos de uso también se clasifican en Simple, Medio y Complejo
  - **Simple**: 3 o menos transacciones.
  - **Medio**: 4 a 7 transacciones
  - **Complejo**: Más de 7 transacciones
- Se cuenta el número de casos de uso en cada categoría, multiplicando ese número por el correspondiente peso y se obtiene el UUCW (Unadjusted Use Case Weigth).
- El UAW + UUCW = *UUP*C El *unadjusted use case points*.

Tipo de Caso de uso	**Peso**
Simple	5
Medio	10
Complejo	15

Métodos Paramétricos

- Se trata de asignar un peso a los factores técnicos o del entorno que pueden influir en el costo de desarrollar el software:
- A Cada factor se le asigna un valor entre 0 y 5 dependiendo de la influencia que tiene sobre el

Factores técnicos

Factor	Descripción	Peso
T1	Sistema distribuido	2
T2	Respuesta Rendimiento Performance	2
T3	Eficiencia del usuario final	1
T4	Procesamiento interno complejo	1
T5	Código reusable	1
T6	Fácil de instalar	0.5
T7	Fácil de usar	0.5
T8	Portable	2
T9	Fácil de cambiar	1
T10	Concurrente	1
T11	Incluye características de seguridad	1
T12	Provee acceso a terceras partes	1
T13	Se requerien entrenamiento especial	1

Factores del entorno

Factor	Descripción	Peso
T1	Familiar con RUP	1,5
T2	Experiencia en la aplicación	0,5
T3	Experiencia en objetos	1
T4	Buena capacidad de análisis	0,5
T5	Motivación	1
T6	Requerimientos estables	2
T7	Trabajadores part - time	-1
T8	Lenguaje de programación	-1

Métodos Paramétricos

- *El **TechnicalComplexity Factor (TCF)*** es calculado en base a multiplicar cada factor de la tabla 1 por su peso y luego sumar todos estos valores para obtener el llamado *TFactor*.
  - Finalmente se aplica la siguiente fórmula: TCF = 0.6 + (.01*TFactor)

- *El **Environmental Factor (EF)*** es calculado en base a multiplicar cada factor de la tabla 2 por su peso y luego sumar todos estos valores para obtener el llamado *Efactor*.
  - Finalmente se aplica la siguiente fórmula: EF = 1.4+(-0.03*EFactor)

- El ***adjusted use case points (UCP)*** se calcula por la siguiente formula:
  - UCP = UUCP*TCF*EF
  - Karner propone un valor de 20 horas /hombre por UCP para cada proyecto
    - UCP * 20 HH = Costo en HH del proyecto
  - *Enfoque de Kirstein Rubi (2001):*
    - *15 a 30 hs x UCP*