Universidad
Rey Juan Carlos

Escuela de Informática del Ejercito
XL Curso DIM – 2019 / 2020

Juan Manuel Vara
juanmanuel.vara@urjc.es
@jmvara

Dirección de Proyectos
Estimación de Proyectos Software

- **Estimación de Proyectos Software**
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de Estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

- **Estimación de Proyectos Software**
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empiricos
  - Métodos de estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

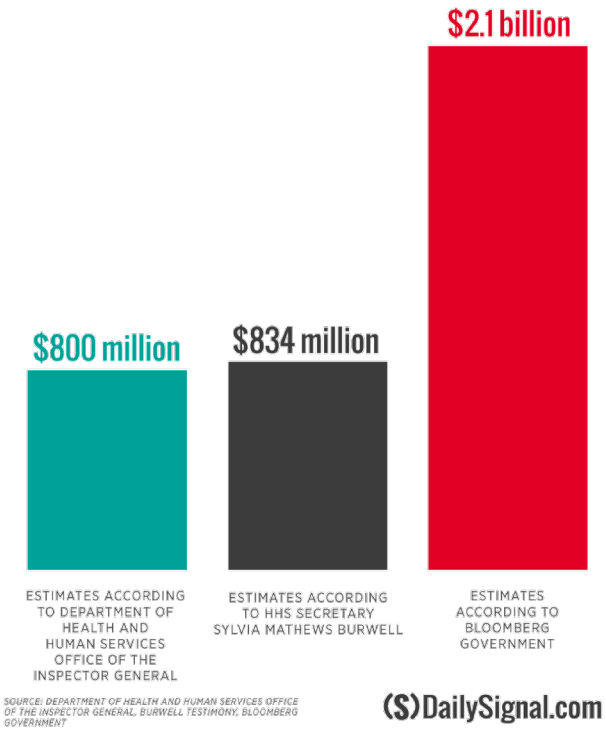
Origen de la estimación

RESOLUTION

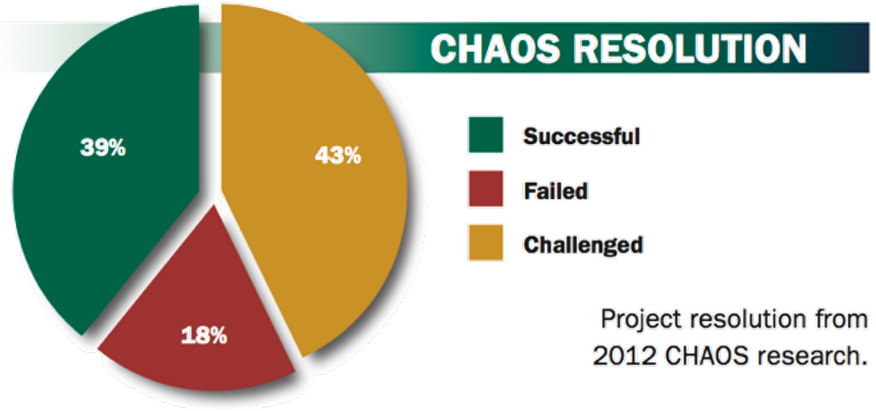
	2004	2006	2008	2010	2012
Successful	29%	35%	32%	37%	39%
Failed	18%	19%	24%	21%	18%
Challenged	53%	46%	44%	42%	43%

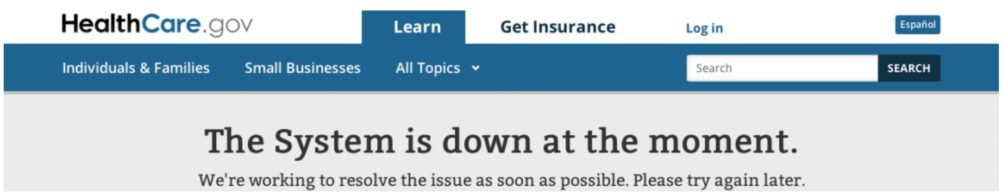
Project resolution results from CHAOS research for years 2004 to 2012.

Cost of HealthCare.gov



http://dailysignal.com/2016/03/23/in-five-charts-how-obamacare-has-worked-past-six-years/





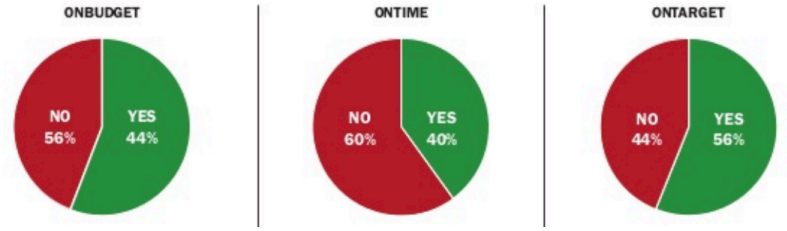
"I'm going to try and download every movie ever made, and you're going to try to sign up for Obamacare, and we'll see which happens first" – Jon Stewart challenging Kathleen Sebelius (former Secretary of Health and Human Services) to a race.

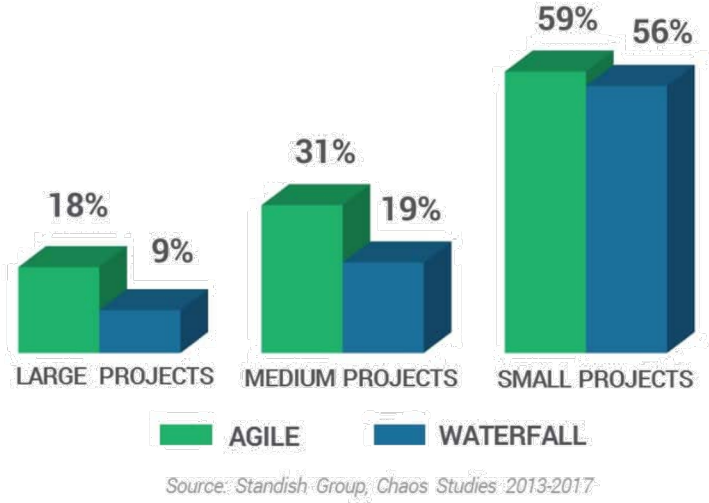
Origen de la Estimación

TRADITIONAL RESOLUTION FOR ALL PROJECTS

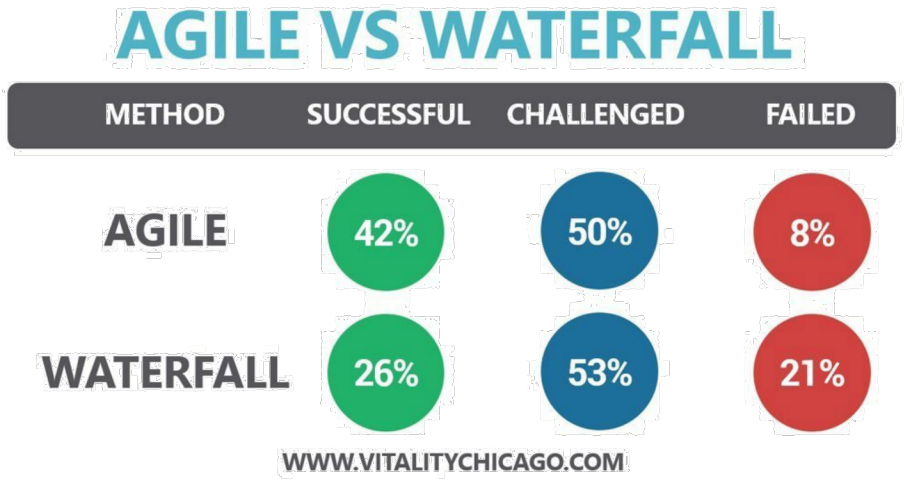
	2011	2012	2013	2014	2015
SUCCESSFUL	39%	37%	41%	36%	36%
CHALLENGED	39%	46%	40%	47%	45%
FAILED	22%	17%	19%	17%	19%

The Traditional resolution of all software projects from FY2011-2015 within the new CHAOS database.





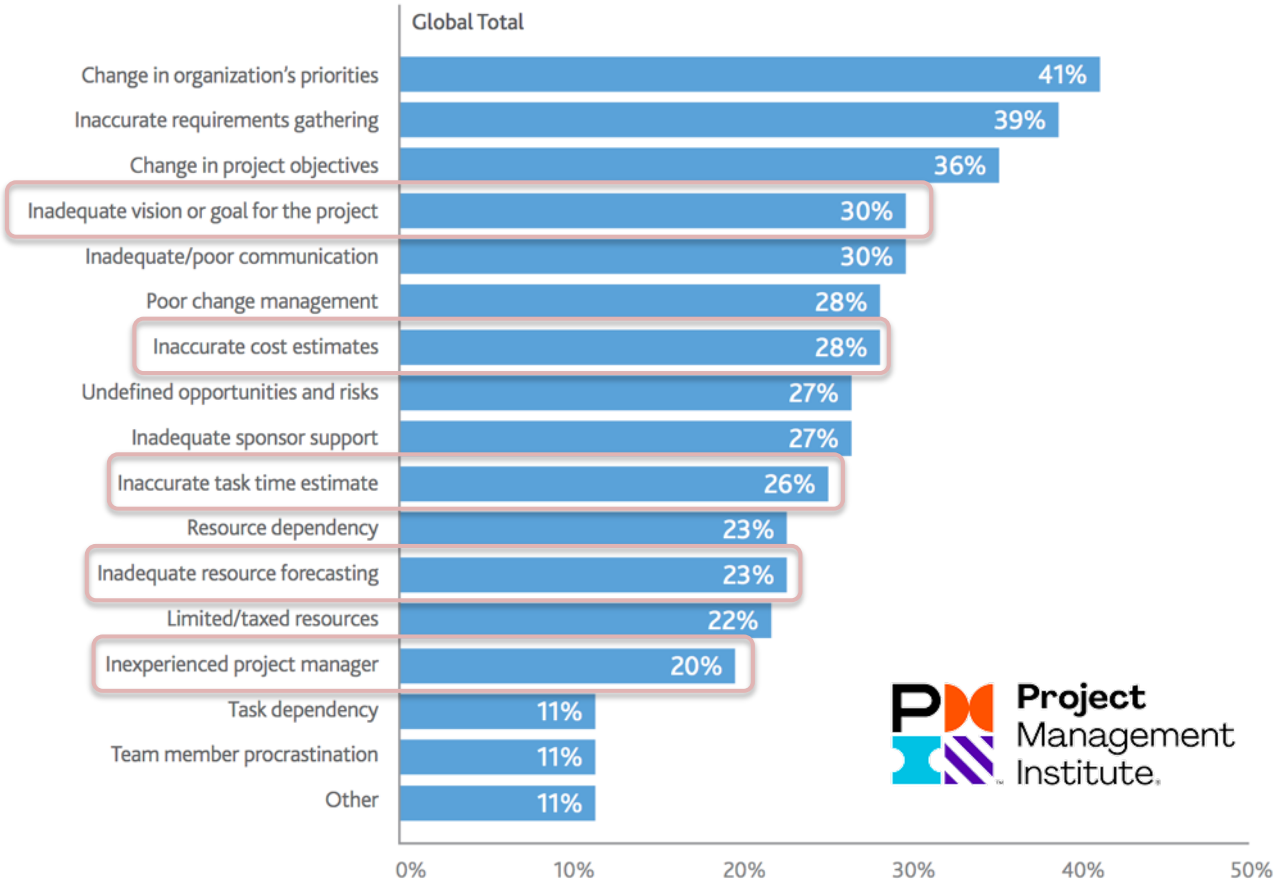
Source: Standish Group, Chaos Studies 2013-2017



SIZE	METHOD	SUCCESSFUL	CHALLENGED	FAILED
All Size Projects	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Large Size Projects	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Medium Size Projects	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Small Size Projects	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

Origen de la estimación

Of the projects started in your organization in the past 12 months that were deemed failures, what were the primary causes of those failures? (Select up to three.)



PMI's Pulse of the Profession 2017

Feedback and insights from 3,234 project management professionals

Top 5 Causes of Troubled Projects

1
REQUIREMENTS
Unclear, lack of agreement, lack of priority, contradictory, ambiguous, imprecise

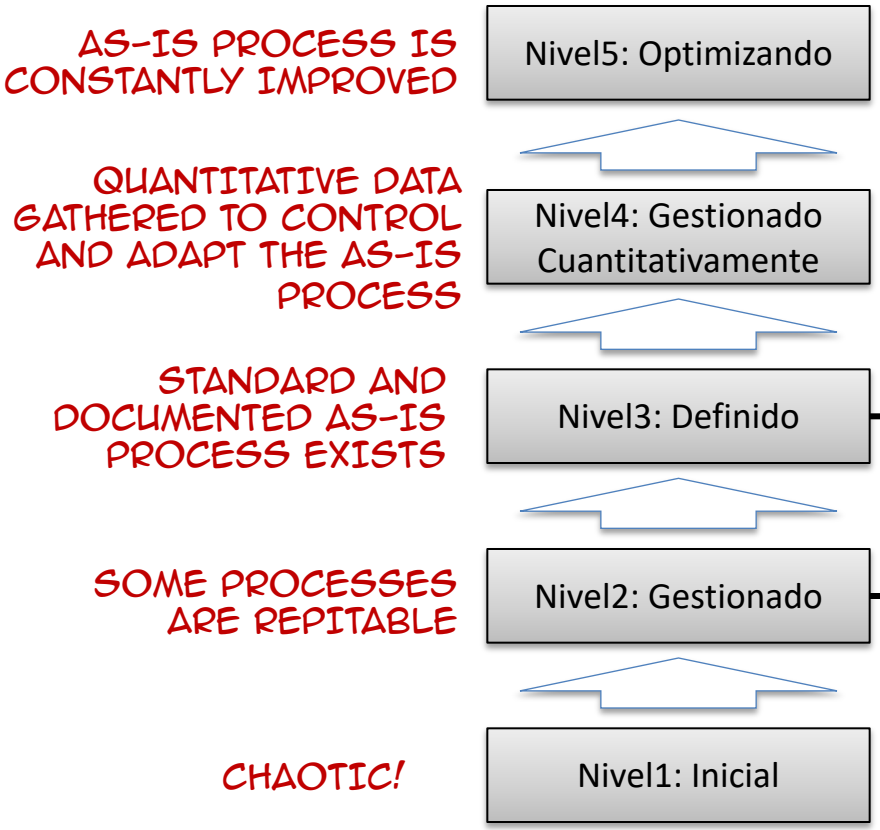
2
RESOURCES
Lack of resources, resource conflicts, turnover of key resources, poor planning

3
SCHEDULES
Too tight, unrealistic, overly optimistic

4
PLANNING
Based on insufficient data, missing items, insufficient details, poor estimates

5
RISKS
Unidentified or assumed, not managed

CMMI-DEV 1.2
Madurez de las organizaciones

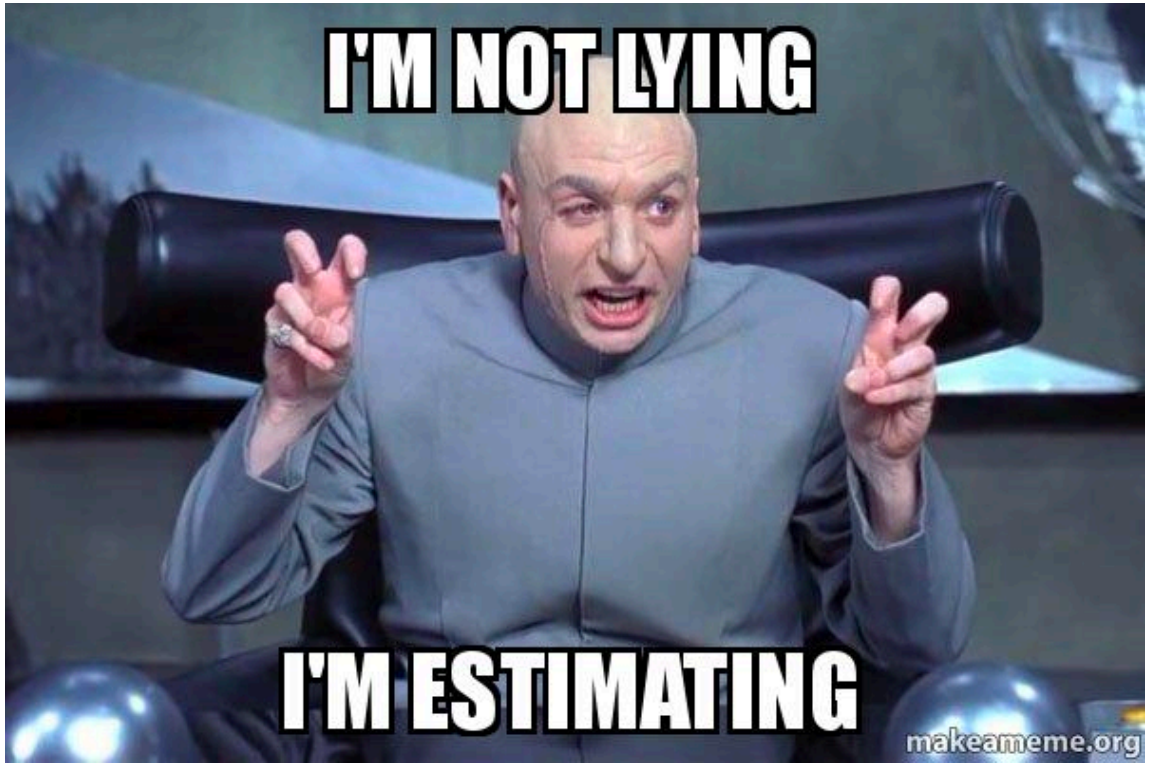


ÁREAS DE PROCESO

- Requirements Development
- Technical Solution
- Product Integration
- Verification
- Validation
- Organizational Process Focus
- Organizational Process Definition
- Organizational Training
- Integrated Project Management
- Risk Management
- Decision Analysis and Resolution

- Requirements Management
- Project Planning
- Project Monitoring and Control
- Supplier Agreement Management
- Measurement and Analysis
- Process and Product Quality Assurance
- Configuration Management

- **Entradas para la planificación**
  - Esfuerzo humano necesario (persona-mes)
  - Duración del proyecto (meses)
  - Coste del proyecto (€)



¿Qué es la estimación?

"Una estimación es una predicción de cuanto tiempo durará o costará un proyecto."

McConell, 2006

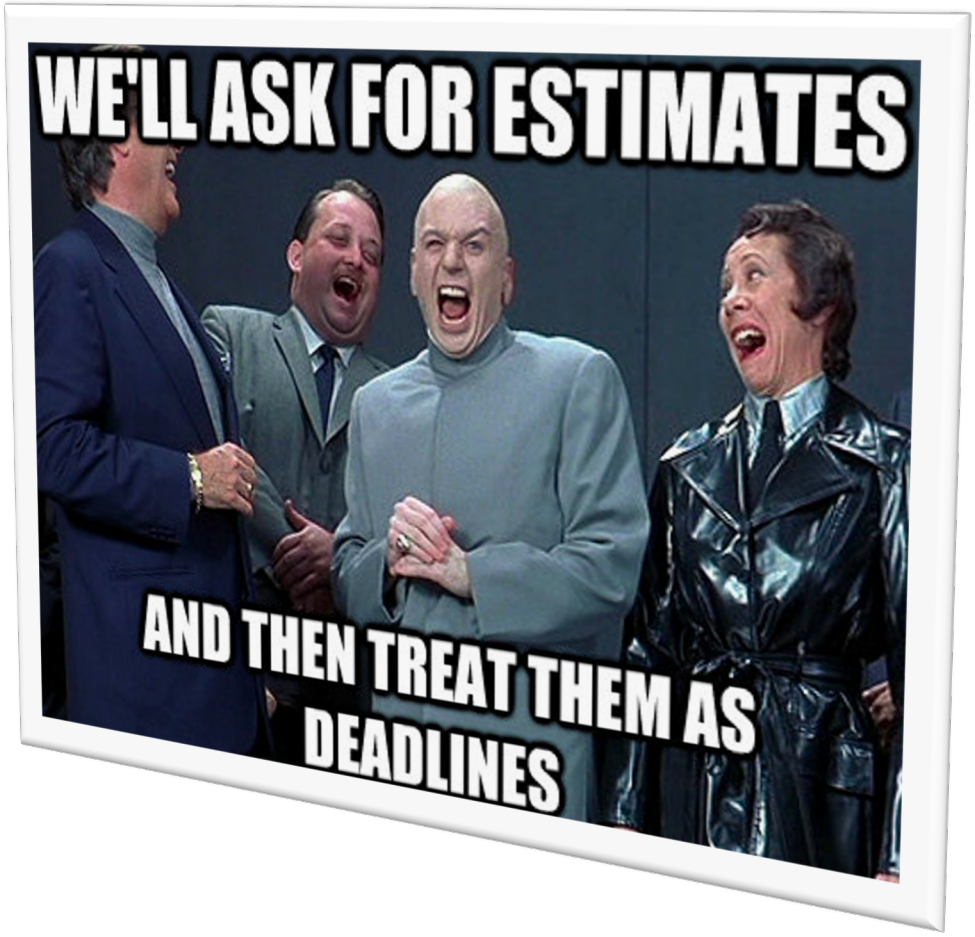
1. A tentative evaluation or rough calculation.
2. A preliminary calculation of the cost of a project.
3. A judgment based upon one's impressions; opinion.

(Source: *The American Heritage Dictionary*, Second College Edition, 1985.)

Estimar o … Planificar?

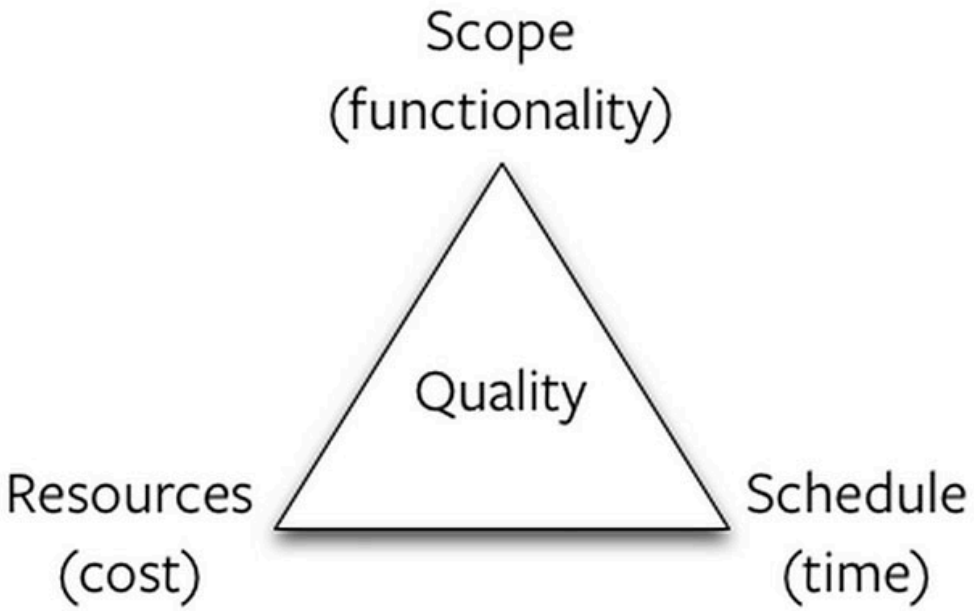
- **Planificar VS Estimar**
  - Proceso sesgado que persigue determinados **objetivos**

    VS
  - Proceso **analítico** que persigue precisión y no unos resultados previamente definidos
- **Estimaciones "a medida"**
  - Proporcionar una planificación razonada que se ajuste al presupuesto establecido
  - Nos están pidiendo una estimación o un plan para alcanzar un objetivo ¿?



- La estimación como base de la planificación
  - El plan debería ser realista, reconocer posible gap estimación ⇆ objetivos

Estimaciones realistas | Negociación



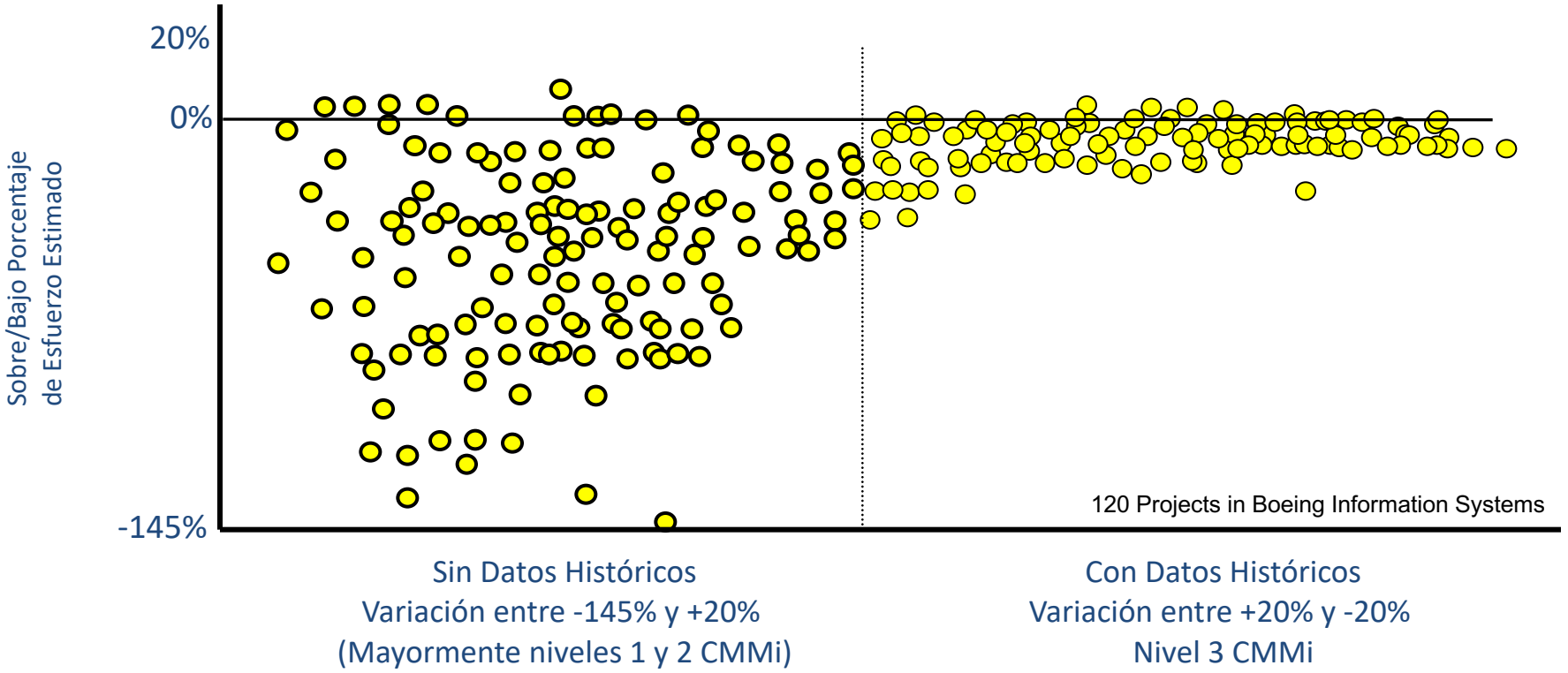
http://blog.resourceguruapp.com/smart-project-negotiating-strategies/

¿Qué es la estimación? - II

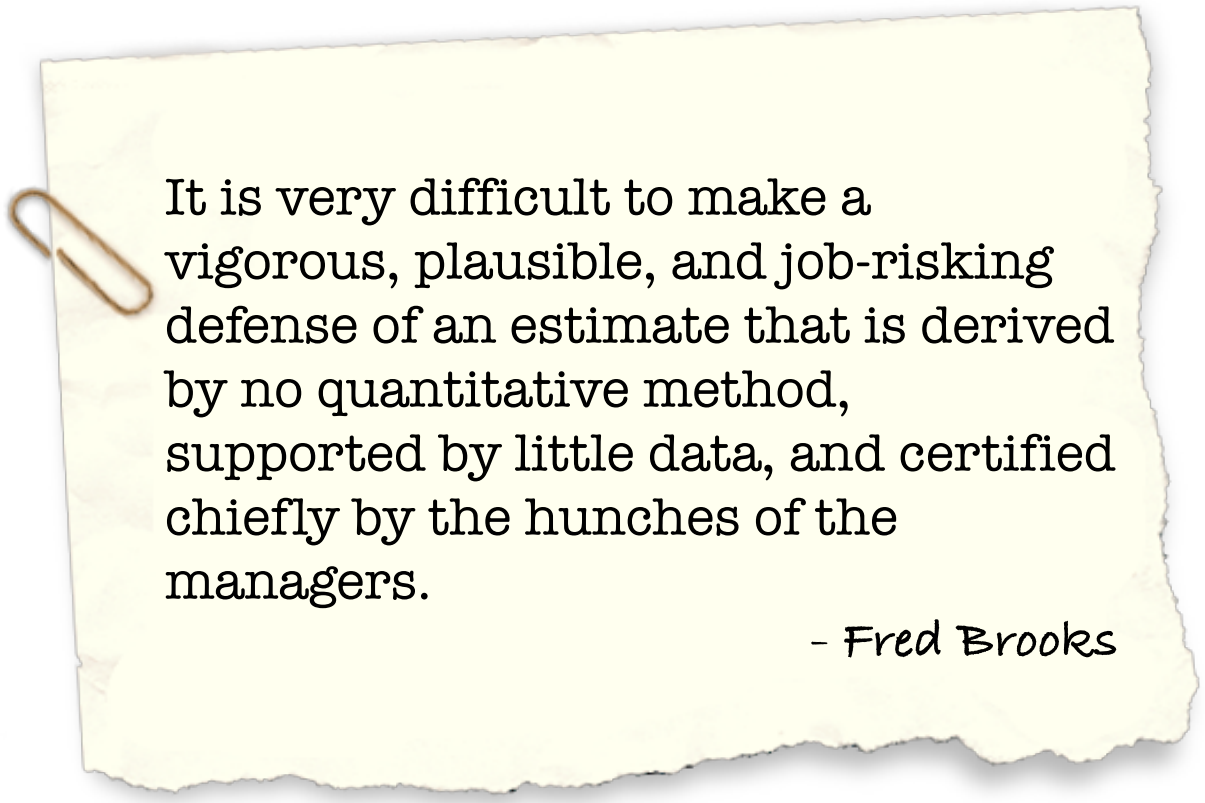
- **Se pretende responder a dos preguntas**
  - ¿Cuánto costará?
  - ¿Qué plazo de tiempo requerirá su desarrollo?
- **Es un proceso continuo, con constantes refinamientos y mejoras, más que una actividad puntual**
  - Los métodos actuales dependen de la cantidad de información disponible.
  - A medida que se avanza en el proyecto, se obtiene mas información y más fiable

Estimación basada en datos históricos

La exactitud de la estimación crece con los datos históricos



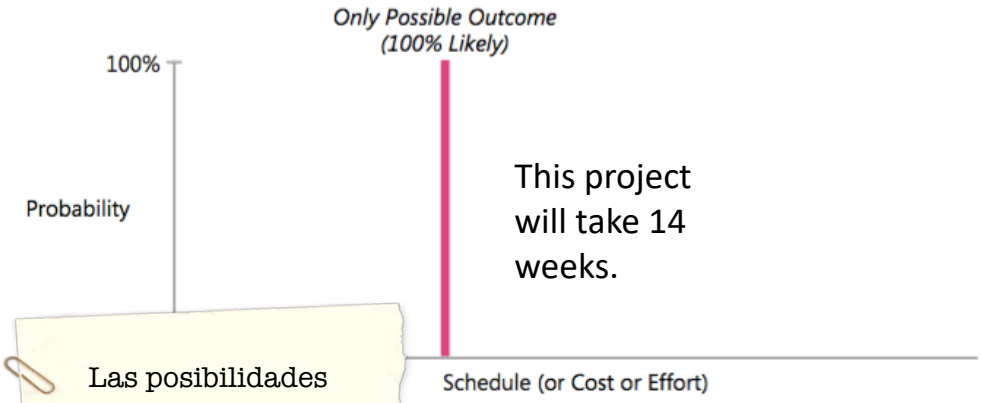
Hasta ahora …


It is very difficult to make a vigorous, plausible, and job-risking defense of an estimate that is derived by no quantitative method, supported by little data, and certified chiefly by the hunches of the managers.
- Fred Brooks

Probabilidad de las estimaciones

- Algunos tratarán de mejorar sus estimaciones para que la desviación de los resultados de sus proyectos se muevan en valores próximos al ±5% respecto a los resultados estimados en lugar de valores ±10%
- Otras organizaciones simplemente aspirarán a realizar estimaciones cuyos ratios de error no superen el 100%

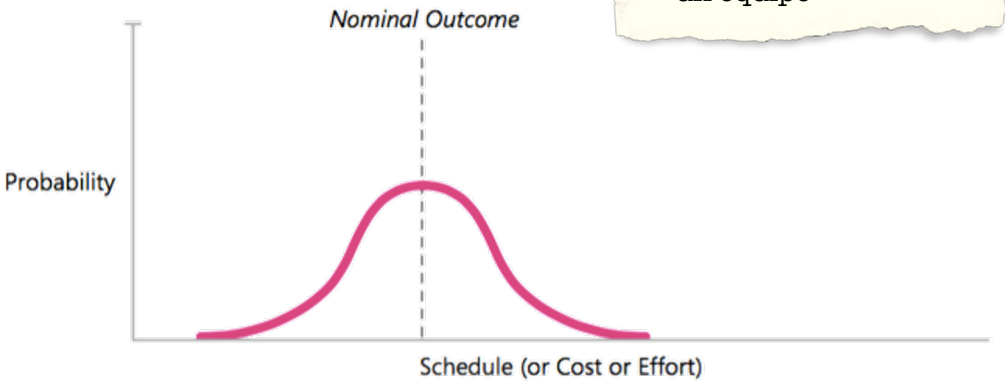
Probabilidad de las estimaciones

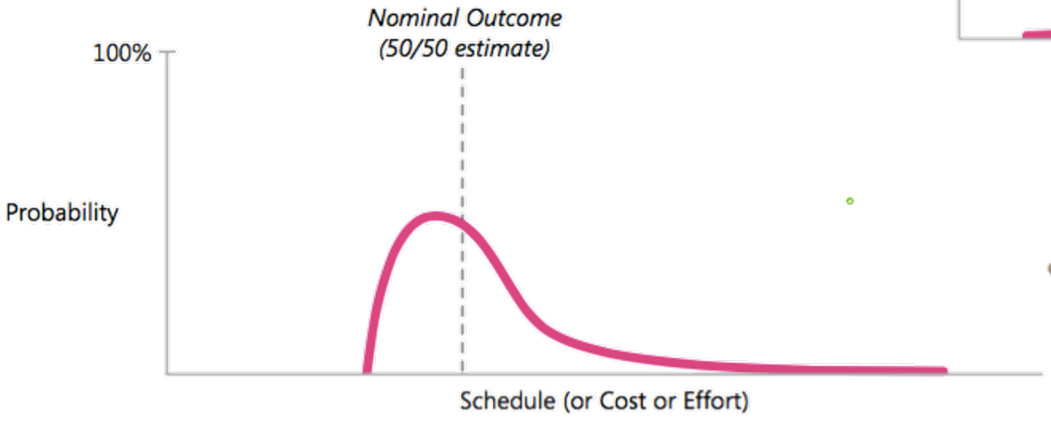


Las posibilidades de acertar al 100% son muy remotas

[McConnell, 2006]

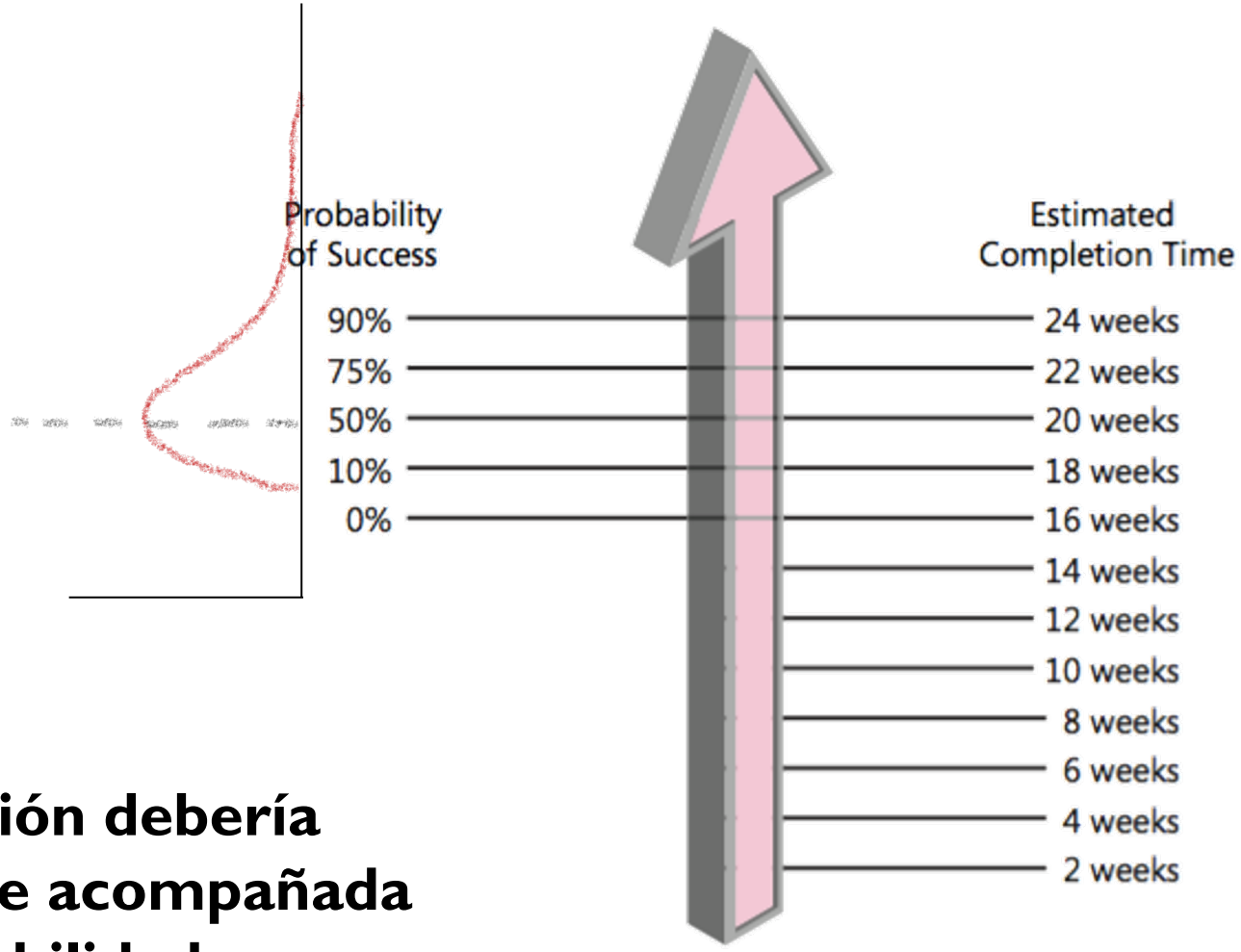
Hay límites en cuanto a cuan eficiente puede ser un equipo





... pero no hay límites en cuanto a la aparición de problemas

Probabilidad de las estimaciones



Una estimación debería venir siempre acompañada de una probabilidad

[McConnell, 2006]

Cuestionario

- **Para cada pregunta, elija un rango de valores que, en su opinión, le asegure una tasa de aciertos ≥ 90%**
  - Defina rangos suficientemente amplios …

(Este cuestionario pretende valorar habilidades de estimación, no habilidades de investigación)

Cuestionario

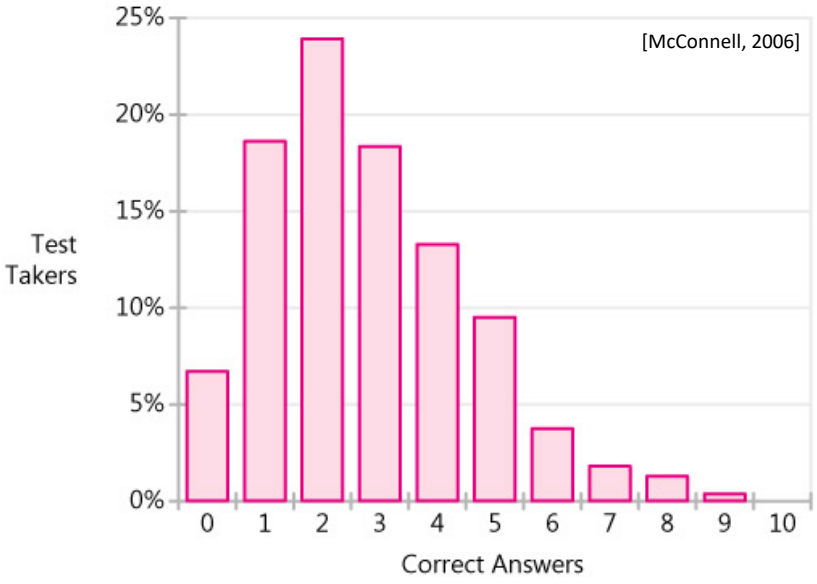
Respuesta (min)	Respuesta (max)	Pregunta
		Temperatura de la superficie del Sol
		Latitud de Shanghai
		Superficie de Asia
		Año nacimiento de Alejandro Magno
		Taquilla mundial de “Titanic”
		Kilómetros de costa Española
		Libros publicados en España en 2018
		Ballena azul más pesada
		Habitantes de Budapest
		Gasto público en España en 2019

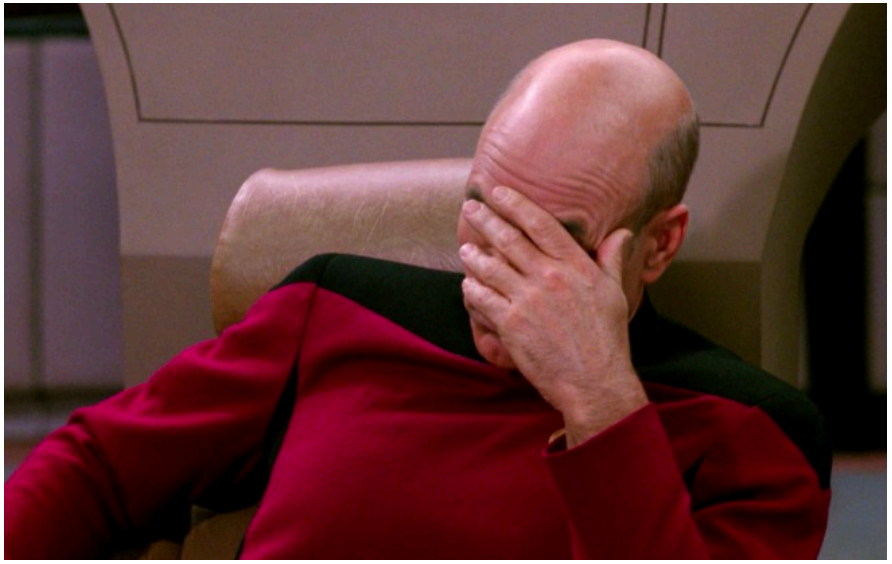
Cuestionario

Respuesta	Pregunta
6000 °C	Temperatura de la superficie del Sol
31° Norte	Latitud de Shanghai
44.390.000 km^2	Superficie de Asia
356 aC	Año nacimiento de Alejandro Magno
$2.186.772.302	Taquilla mundial de “Titanic”
7905 kms	Kilómetros de costa Española
81.228	Libros publicados en España en 2018
190 T	Ballena azul más pesada
1.757.600 personas	Habitantes de Budapest
521.949 millones €	Gasto público en España en 2019

Rangos de estimación

- ¿Qué hemos aprendido?
  - No proporcione estimaciones con cierto porcentaje de confianza, (cosas como "estoy seguro al 90%") si no dispone de una base cuantitativa de la que derivar su estimación
  - Es probable que sus conocimientos a cerca del proyecto disten muy poco de los que posee sobre la recaudación de Titanic …

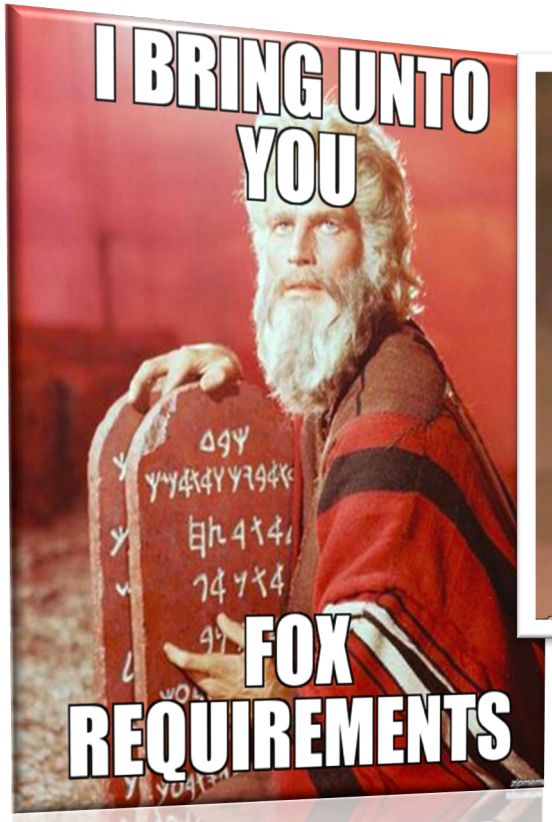


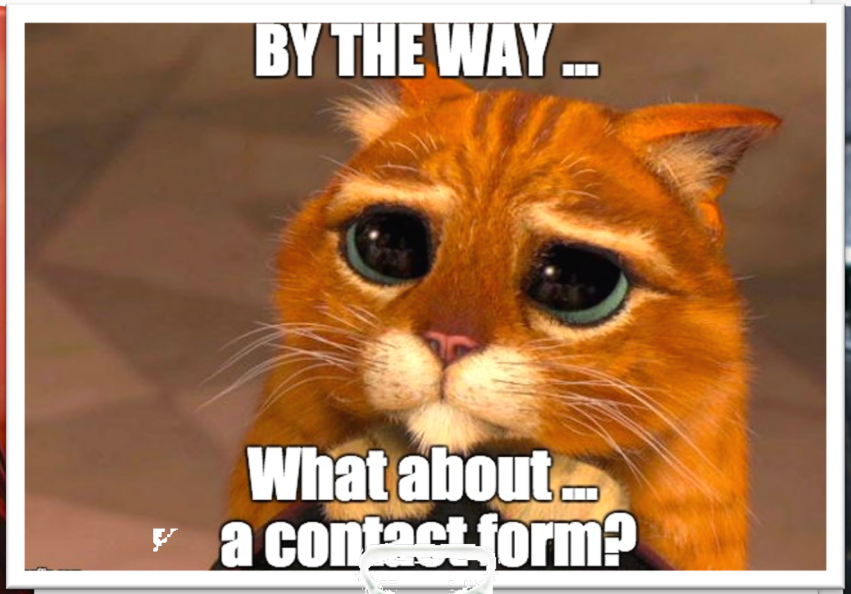


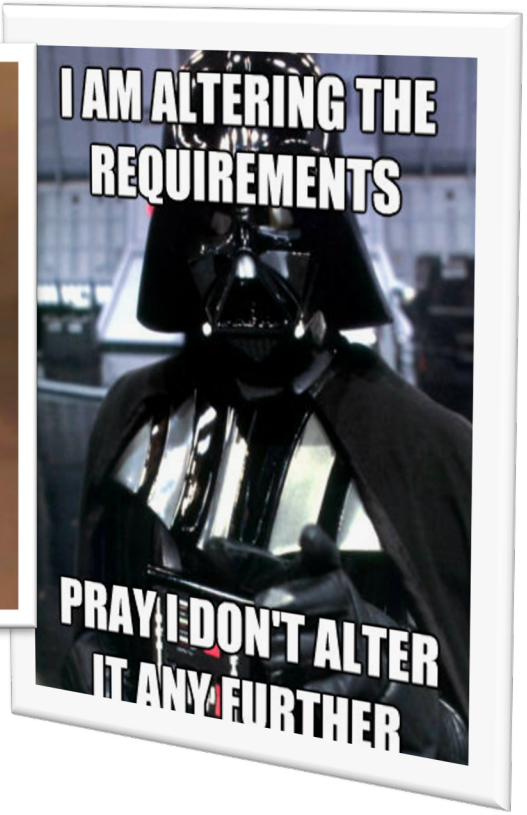
- **El software, frente a otras industrias, es un producto fundamentalmente intelectual**
  - El coste de su producción está dominado por los **gastos de personal**
  - No suele medirse en unidades monetarias.
  - Suelen ser valoraciones, con un cierto error (20%), del esfuerzo y plazos de tiempo.
  - La principal unidad de medición de coste suele ser el número de salarios mensuales o anuales (**personas-mes** o **personas-año**).

- **Razones que dificultan la estimación en proyectos software**
  - Peculiaridades del software.-
    - Es habitual desarrollar un nuevo producto cada vez, empleando distintas técnicas y herramientas.
  - Existe una carencia generalizada de datos sobre proyectos terminados
  - Presiones políticas en la empresa (para disminuir el coste o los plazos necesarios)
  - Actitud habitual de los responsables de la estimación
    - Se espera que esta actividad no tenga un coste

¿Por qué fallan las estimaciones?







- The Cone of Uncertainty can't be narrowed
- Project is often not reestimated

To deal with unstable requirements, consider **project control** strategies instead of or in addition to estimation strategies

- **Hacer estimaciones significa**
  - Suponer lo que ocurrirá: echar un vistazo al futuro
  - Asumir un determinado grado de riesgo como consecuencia de la incertidumbre.
  - Demasiadas variables
  - Pero …
    - Podemos reducir el riesgo si adoptamos una aproximación sistemática

- **Para realizar estimaciones seguras de costes y esfuerzo tenemos varias opciones posibles:**
  - Dejar la estimación para más adelante
  - Basar las estimaciones en proyectos similares ya terminados.
  - Usar técnicas de descomposición
  - Utilizar uno o más modelos empíricos → $d = f(V_i)$

DURACIÓN
ESFUERZO
COSTE

TAMAÑO
LDC – PF)

Combinación de varios métodos

- **Estimaciones eficaces**
  - Personas con mucha experiencia en el ámbito del tipo de proyecto
  - Información fiable sobre el ámbito y requisitos del sistema → Importancia del Análisis del Sistema
  - Datos históricos fiables de proyectos anteriores similares
  - Técnicas de descomposición del problema y del proceso para obtener grupos funcionales
  - Métodos empíricos de estimación para casos particulares

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de Estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

[Piattini, 2008]

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
      - Juicio de expertos
      - Estimación por analogía
      - Estimación por descomposición (↑ ó ↓)
      - Otros: "Precio para ganar" o "Ley de Parkinson"
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de Estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

- Juicio de Expertos
  - Existen técnicas específicas que intentan sistematizar y mejorar la opinión de las distintas personas involucradas en la estimación:
    - Se suele emplear la opinión de más de un experto para obtener una mayor fiabilidad en la estimación.
    - En algunos casos, se calcula la media de los valores ofrecidos por las distintas personas.
      - RIESGO: la estimación puede resentirse mucho por uno o dos valores extremos.
    - Realizar una reunión tan larga como sea necesaria hasta llegar a un consenso.
      - RIESGO: que las personas más influyentes, por su capacidad o por su poder de convencimiento, sean las que determinen el resultado final.
    - Técnica Delphi

[BOEHM, 1981]

- **Juicio de Expertos – Técnica Delphi**

[BOEHM, 1981]

1. Un coordinador proporciona a cada experto una especificación del proyecto y un impreso para expresar su opinión.
2. Los expertos rellenan el impreso de forma anónima. Pueden hacer al coordinador preguntas sobre el proyecto pero no pueden intercambiar opiniones entre ellos.
3. El coordinador ofrece a cada experto el valor medio de las opiniones para que la compare con la suya. Se pide realizar una nueva estimación anónima, indicando las posibles razones de la misma.
4. Se repite el proceso de recogida de opiniones hasta que se llega a un consenso en la estimación. No se realizan reuniones en grupo durante todo el proceso.

- **Juicio de Expertos – Técnica Delphi de banda Ancha**
  1. Un coordinador proporciona a cada experto una especificación del proyecto y un impreso para expresar su opinión.
  2. **El coordinador reúne a los expertos para que intercambien puntos de vista sobre el proyecto**
  3. Los expertos rellenan el impreso de forma anónima.
  4. El coordinador ofrece a cada experto el valor medio de las opiniones para que la compare con la suya.
  5. **El coordinador convoca una reunión de grupo para que los expertos discutan las razones de las diferencias entre sus estimaciones**
  6. Se pide realizar una nueva estimación anónima, sin indicar las posibles razones de la misma.
  7. Se rellenan anónimamente los impresos y se repiten los puntos 4, 5 y 6 hasta que se llegue a un consenso.

[BOEHM, 1981]

- **Juicio de expertos**
  - Ventaja
    - Permite contemplar las diferencias entre experiencias anteriores y el proyecto actual, difíciles de evaluar sin recurrir a personas experimentadas.
  - Desventaja
    - Subjetividad e inexperiencia que pueden mostrar las personas a las que se consulta.
      - Contratación como medio de paliar la subjetividad

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
      - Juicio de expertos
      - Estimación por analogía
      - Estimación por descomposición (↑ ó ↓)
      - Otros: "Precio para ganar" o "Ley de Parkinson"
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de Estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

- **Estimación por Analogía**
  - Es un complemento al juicio de expertos.
  - Las personas involucradas no sólo utilizan su experiencia, sino que disponen también de datos de proyectos acabados similares.
  - Se pueden evaluar las diferencias entre el nuevo proyecto y los antiguos y extrapolar su coste.
    - Los ajustes en el coste, esfuerzo o tamaño del nuevo proyecto pueden realizarse de forma lineal (se mantiene aproximadamente la proporcionalidad).
  - Sin embargo, los plazos de tiempo no guardan una relación lineal con el esfuerzo o el tamaño del proyecto.
  - Cuando se dispone de bastantes datos de proyectos terminados, se puede mejorar la analogía.
    - Seleccionando dos proyectos parecidos al actual, uno mayor y otro menor, se puede obtener una mejor estimación interpolando los valores de ambos.

- Para seleccionar el proyecto análogo se deben de tener en cuenta una serie de atributos

ATRIBUTOS PRINCIPALES	VALORES
TIPO DE PROYECTO	Desarrollo, Mantenimiento
TAMAÑO	Puntos Función – Puntos Caso de Uso
OBJETIVOS DEL PROYECTO	Coste, Duración, Esfuerzo y Productividad
PLATAFORMA DE DESARROLLO	Mainframe, MidRange, PC. Multi-Plataforma
TIPO DE LENGUAJE	3GL, 4GL, OO, Scripting
LENGUAJE	Lenguaje de Programación

- Para seleccionar el proyecto análogo se deben de tener en cuenta una serie de atributos

ATRIBUTOS (Otros)	VALORES
UTILIZACIÓN DE METODOLOGÍA	Si / No
TIPO DE METODOLOGÍA	Cascada / Iterativa e Incremental / Ágil
UTILIZACIÓN DE HERRAMIENTA CASE	Si / No
HERRAMIENTA CASE UTILIZADA EN CADA ETAPA DEL CICLO DE VIDA	SINERGY (Gestión de Configuración, DOORS (Gestión de requisitos), TAU (Diseño)
TIPO DE ARQUITECTURA	Cliente/Servidor; Capas; Distribuida; Cloud
ÁREA DE NEGOCIO	Telecomunicaciones, Transporte, Administración Pública, etc.

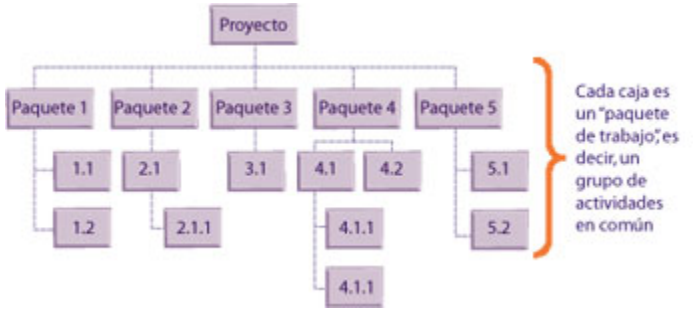
- **Estructura de la BBDD de histórico de proyectos**
  - Datos generales.
    - Datos genéricos del proyecto, como identificador del proyecto, el tipo de proyecto, etc.
  - Datos estimados.
    - Se refieren a los datos que se estiman al inicio del proyecto, como el esfuerzo, etc.
  - Datos reales.
    - Los mismos datos que se estimaron al inicio del proyecto se calcularán nuevamente a lo largo del desarrollo del proyecto
  - Diferencias.
    - Estos datos surgen de la diferencia entre los datos estimados y los datos reales.

- **Estimación por Analogía**
  - Ventaja
    - Basada en la experiencia real (no sólo en la subjetiva) de los proyectos.
  - Desventaja
    - Es difícil conocer realmente el grado de similitud del proyecto que se estima con el terminado elegido.
    - Su éxito depende de contar con una buena colección de datos de proyectos (por otra parte, esto es esencial para el buen funcionamiento de todos los métodos de estimación).

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
      - Juicio de expertos
      - Estimación por analogía
      - Estimación por descomposición (↑ ó ↓)
      - Otros: "Precio para ganar" o "Ley de Parkinson"
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de Estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

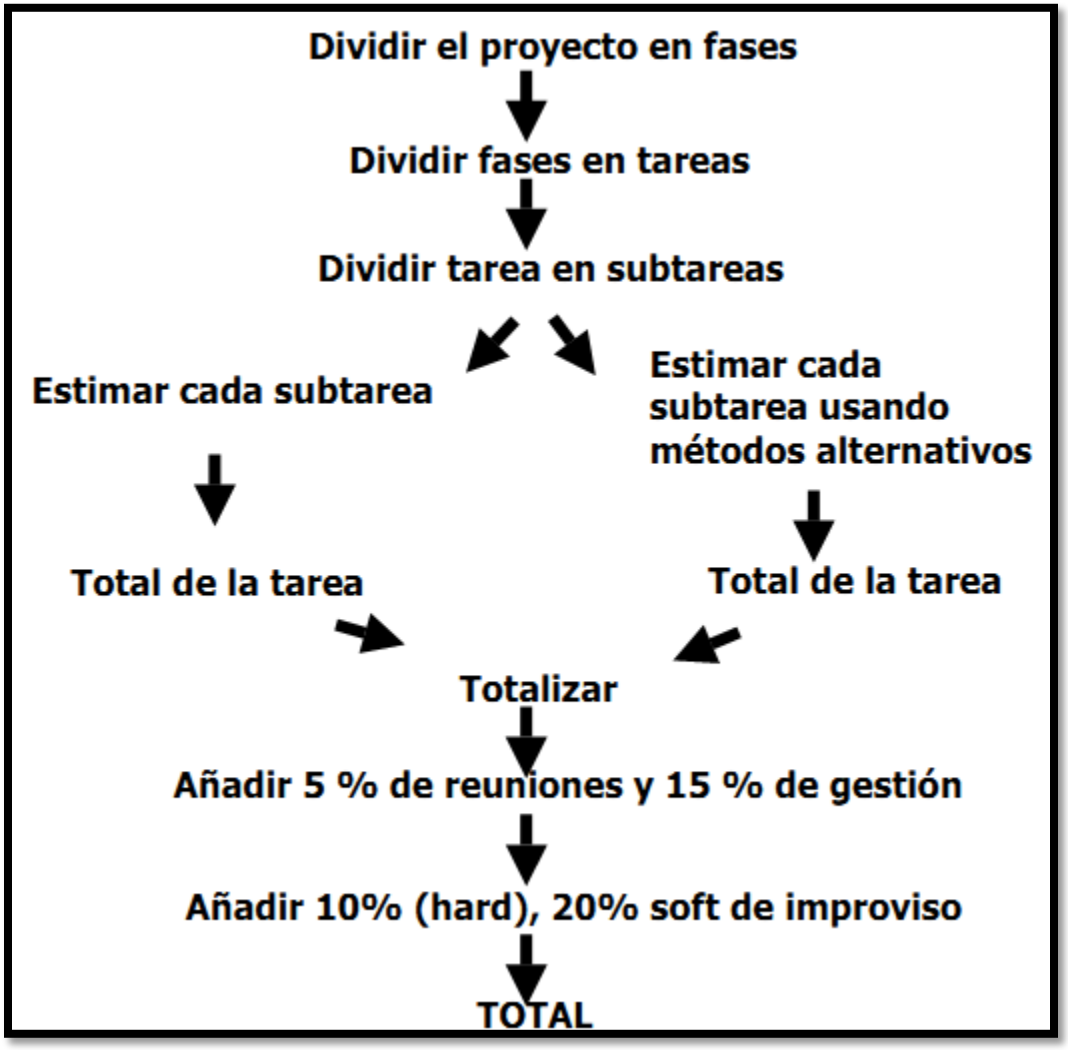
- **Estimación por descomposición**
  - El responsable de cada componente del software estima el coste de su desarrollo.
  - La estimación se calcula mediante la suma de las cantidades parciales (enfoque bottom-up).
  - Para poder aplicarla con una mínima eficacia:
    - Se necesita disponer de un diagrama de descomposición del producto (WBS del producto) que representa la jerarquía del producto.
    - Suele complementarse con un diagrama de descomposición de actividades (WBS del trabajo) que indica la jerarquía de tareas.
      - De esta manera se asegura que actividades como la integración de componentes, o la gestión de la configuración, queden reflejadas en la estimación del coste.

También existen aproximaciones top-down



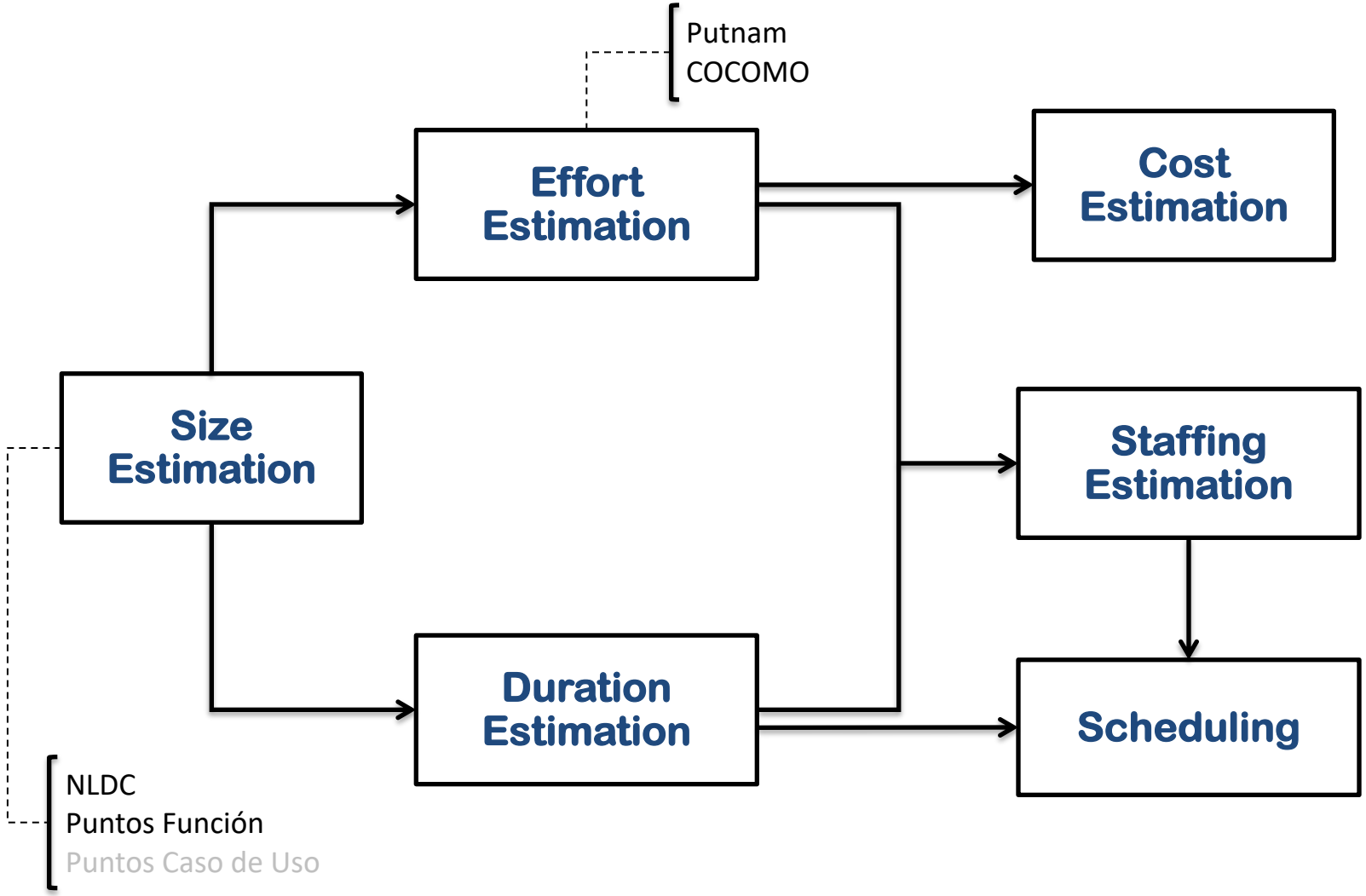
- **Estimación por descomposición**
  - Ventajas
    - Obliga a comprender mejor la tarea a desarrollar
    - Permite a cada componente del equipo de desarrollo planear su trabajo, asegurando el compromiso personal de cada uno con la estimación obtenida
  - Desventajas
    - Proceden de la dificultad para considerar dos posibles fuentes adicionales de coste (Boehm, 1981)
      - Actividades relacionadas con el proyecto que no suelen incluirse en la definición del mismo.
      - Actividades no relacionadas con el proyecto

- Estimación por descomposición



- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
      - **Estimación de Tamaño: NLDC, Puntos Función**
      - Estimación de Esfuerzo, Duración y Coste
    - Métodos Algorítmicos o Empíricos
  - Métodos de Estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

Método y unidad de medición del tamaño



- **Estimación de tamaño**
  - Número de Líneas de Código (NDLC ó LDC)
    - Ventajas
      - Existen datos históricos (tooling)
      - Esfuerzo por LDC parece ser bastante constante
      - Permite comparar proyectos

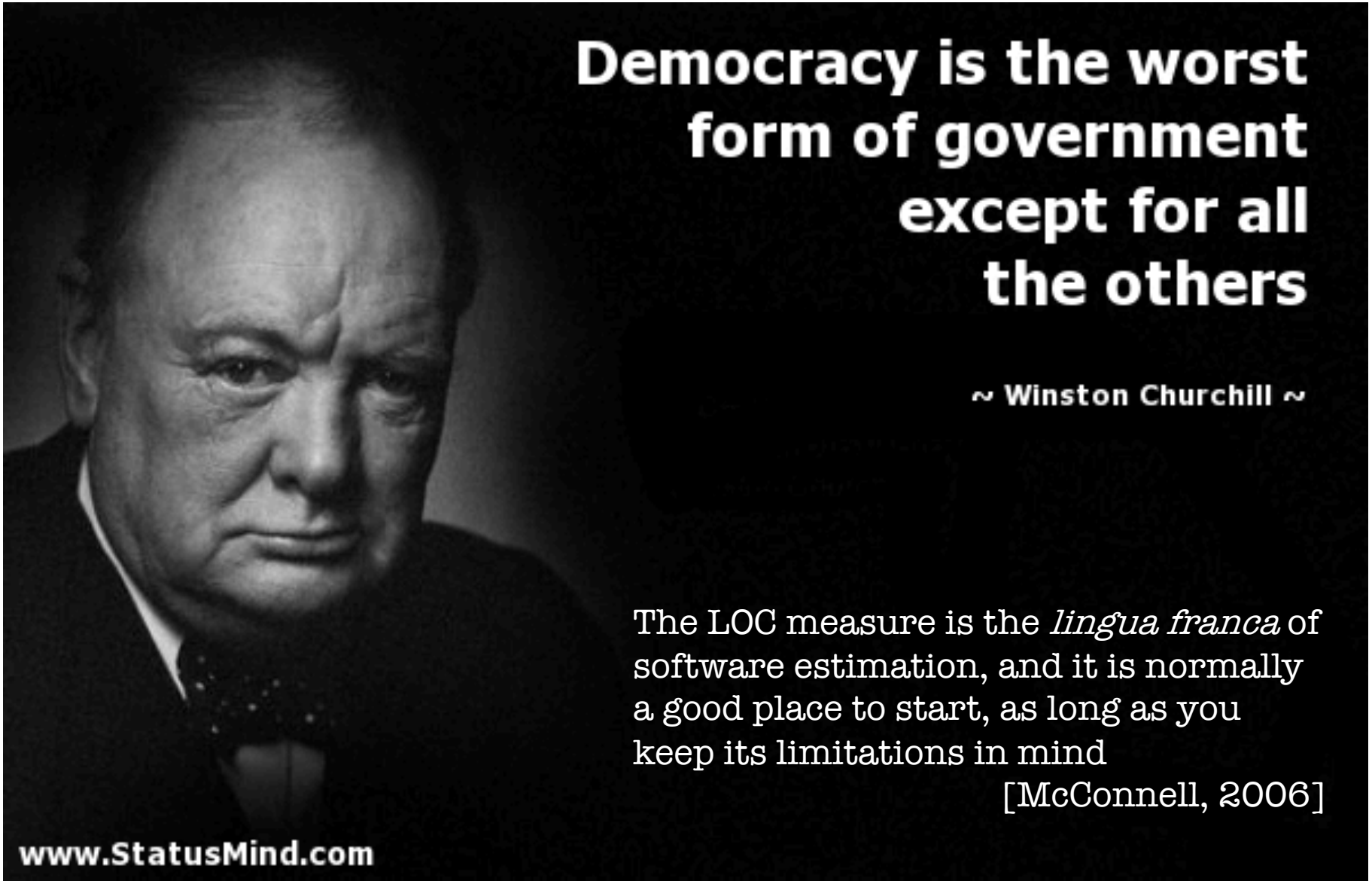
[Sommervile, 2005]

- **Estimación de tamaño**
  - Número de Líneas de Código (NDLC ó LDC)
    - Desventajas
      - Tiempo para análisis, diseño, etc.
      - Falta una definición universal de qué es una LDC
      - Programadores verbosos son mejores??
      - Diferencias entre los distintos lenguajes: COBOL, FORTRAN: 1 tarjeta = 1LDC
      - Métricas multidimensionales ¿?

[Sommervile, 2005]

	Análisis	Diseño	Implementación	Pruebas	Documentación
Código Ensamblador	3 semanas	5 semanas	8 semanas	10 semanas	2 semanas
Lenguaje de Alto Nivel	3 semanas	5 semanas	4 semanas	6 semanas	4 semanas
	Tamaño	**Tiempo**			**Productividad**
Código Ensamblador	5000 líneas	**28 semanas**			**714 líneas/mes**
Lenguaje de Alto Nivel	1500 líneas	**20 semanas**			**300 líneas/mes**

Estimación de tamaño: NDLC



The LOC measure is the *lingua franca* of software estimation, and it is normally a good place to start, as long as you keep its limitations in mind

[McConnell, 2006]

- **Estimación de tamaño**

[Sommerville, 2005]
[Piattini, 2008]

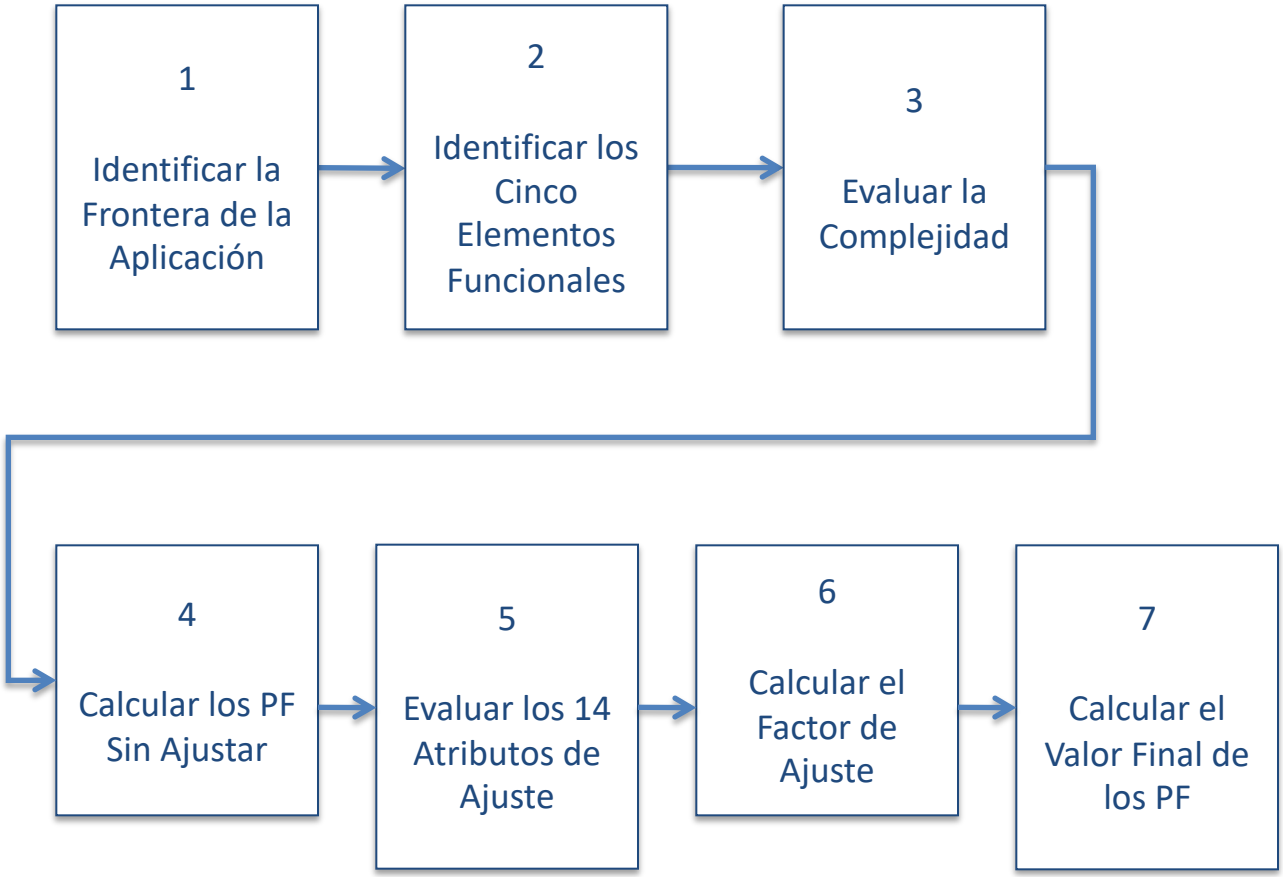
  - Puntos Función
    - El método de estimación mediante los puntos de función ha sido denominado FPA (Function Point Analisys, - Análisis de Puntos Función).
    - Es la técnica algorítmica de estimación más conocida
    - Miden el tamaño lógico o funcional de los proyectos o aplicaciones software basado en los requerimientos funcionales del usuario
      - Tamaño. Un punto función (PF) es una medida sintética del tamaño de un programa que pretende medir su funcionalidad y no el NLDC.
      - Aplicaciones. Mide software, no considera hardware, ni la administración del proyecto, ni la documentación, etc.
      - Funcionalidad. Capacidad del software para que un usuario pueda realizar transacciones (lectura, escritura, etc.) y guardar datos.
      - Usuario. Quién lo va a utilizar y no quién lo desarrollará o diseñará.
    - Utiliza un modelo paramétrico (lista de parámetros) y estaba orientado inicialmente a aplicaciones de gestión.
    - Todas las variedades de puntos de función se apoyan en datos que implican, preferentemente, la existencia de una especificación más o menos formal.

- **Puntos Función**
  - Enfoque en la funcionalidad, independiente de la tecnología
    - Una vez establecida la funcionalidad, podemos escoger tecnología
  - Simple
    - Establecer tamaño de productos de KLDC en pocas horas
  - Basada en los requisitos de usuario
    - El usuario podría llegar a entender la estimación sin ser un experto en sistemas
    - Podríamos llegar a establecer el tamaño del sistema con la ERS

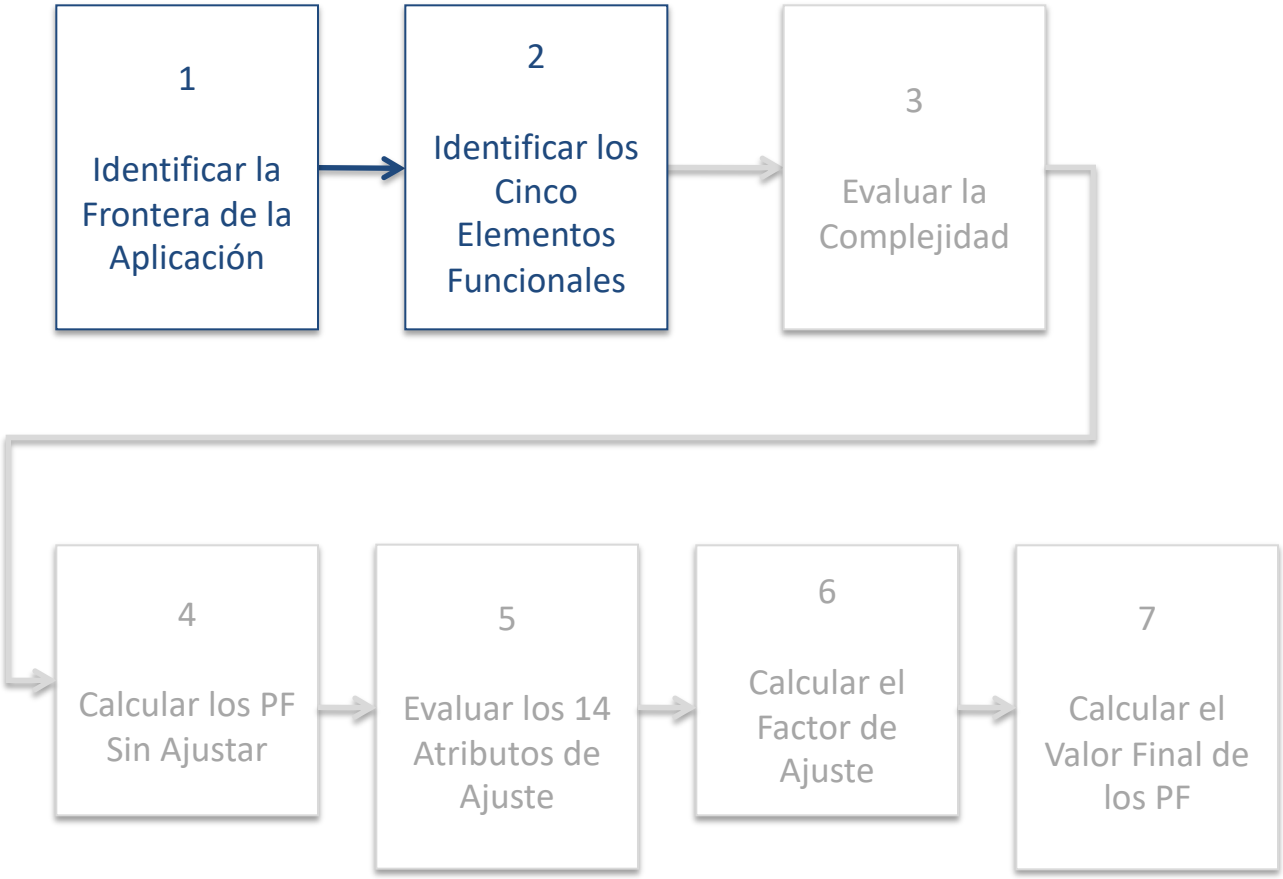
Estimación de tamaño: Puntos Función

En sus comienzos	1979 – Allan Albrecht crea los FPA
	1984 – Primer manual formal sobre FPA (IBM)
	1987 – El gobierno británico adopta FPA (una adaptación llamada MkII)
Decada de los 90	Se extiende a Australia, Brasil, Italia, Japón y Sudáfrica
	Aparecen algunas variaciones – COSMIC-FFP, NESMA FPA, FiSM FSM, Use Case Points, Early and Quick Function Points, etc.
	1994 – IFPUG publica la versión 4 del manual de medición
Siglo XXI	Se extiende a Corea, India y China
	1999 - IFPUG publica la versión 4 del manual de medición
	2003 – IFPUG se convierte en el estándar ISO 20926 (no se consideran los factores de ajuste)
	2004 – IFPUG publica la versión 4 del manual de medición

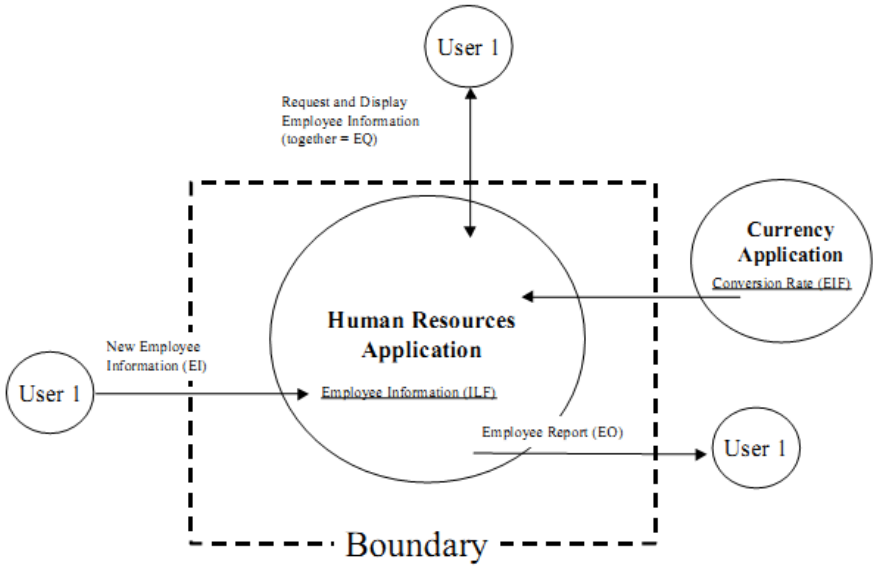
- Proceso (Método Estándar de IFPUG)



- Proceso (Método Estándar de IFPUG)

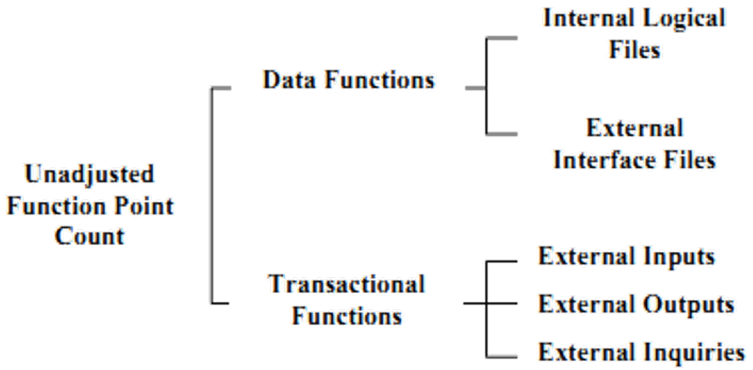


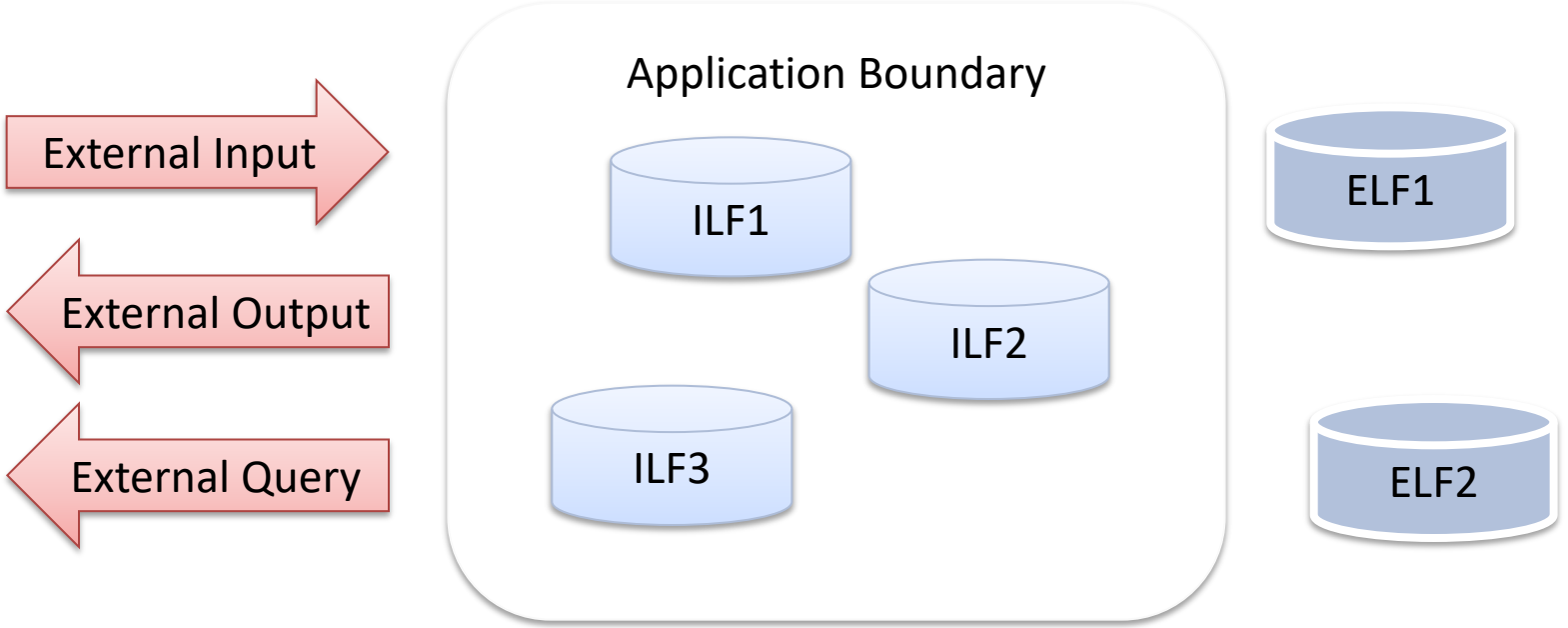
- **Identificar los límites de la aplicación**
  - Definen lo externo a la aplicación
  - Se determinan en función del punto de vista del usuario



Identificar los límites de la Aplicación

- Identificar los Cinco Elementos Funcionales



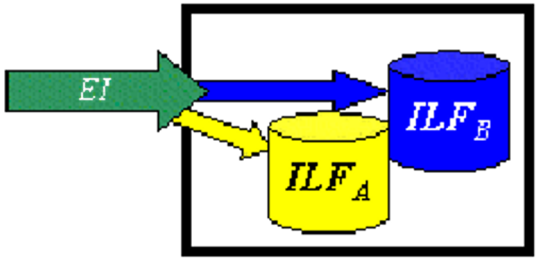


- **Identificar los Cinco Elementos Funcionales**
  - Fichero Lógico Interno (ILF)
    - Grupo de datos lógicamente relacionado y mantenido dentro de los límites de la aplicación
    - Objetivo: almacenar datos mantenidos por uno o más procesos elementales
    - Ejemplos
      - Datos de empleados
      - Preferencias de usuario

ENTIDADES E/R

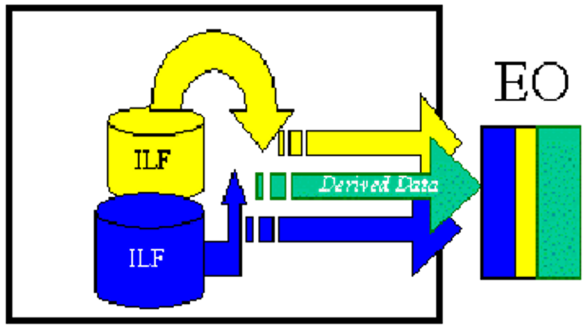
- **Identificar los Cinco Elementos Funcionales**
  - Fichero Lógico Externo (ELF)
    - Grupo de datos relacionados y referenciados por el sistema pero mantenidos desde otra aplicación
    - Objetivo: almacenar datos referenciados mediante uno o más procesos elementales (o acciones)
    - Ejemplo:
      - Las tasas de cambio son información que utiliza la aplicación de Gestión de RRHH … pero mantenida por la aplicación de contabilidad

- **Identificar los Cinco Elementos Funcionales**
  - Entrada Externa
    - Proceso elemental que procesa datos, o información de control, que vienen del exterior de la aplicación.
    - Objetivo: mantener uno más archivos lógicos internos y/o alterar el funcionamiento del sistema
      - Si no son información de control que altere el funcionamiento, dichos datos servirán para actualizar al menos un ILF
    - Ejemplo
      - Añadir datos de empleados

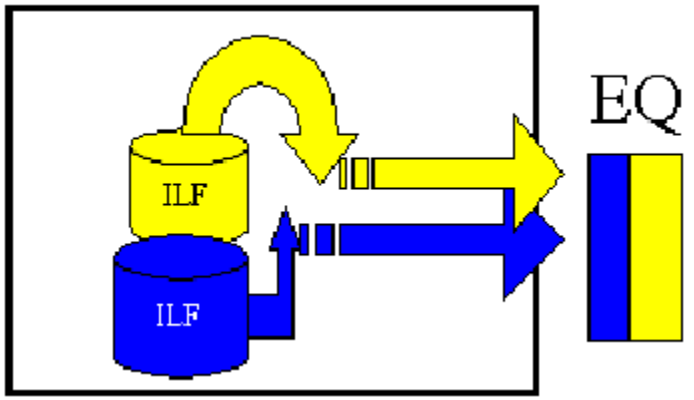


- Recoge los datos introducidos por pantalla, comprueba formulario y genera mensaje de error si el formulario no está relleno correctamente
- Guarda los datos introducidos en el ILF correspondiente

- **Identificar los Cinco Elementos Funcionales**
  - Salida Externa
    - Proceso elemental que envía datos o información de control al exterior de la aplicación.
    - Objetivo: extraer datos derivados para proporcionárselos al usuario
    - Contiene al menos una fórmula o cálculo matemático ó crea datos derivados a partir de ILFs/ELFs ó mantiene uno o más ILFs ó modifica el comportamiento del sistema.
    - Ejemplo
      - Envío de nómina a final de cada mes
      - Producción de informes de los datos de empleados contenidos en la aplicación



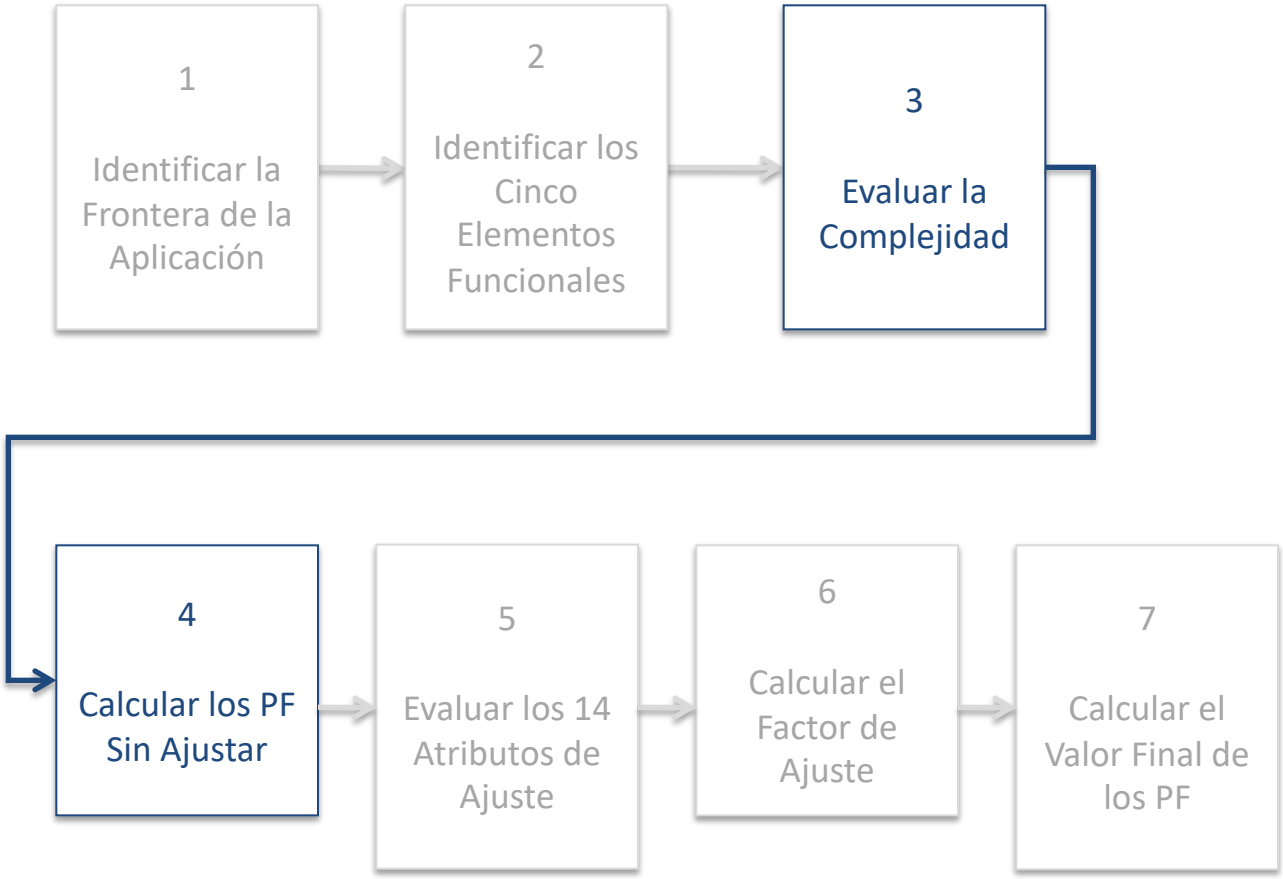
- **Identificar los Cinco Elementos Funcionales**
  - Consulta Externa
    - Proceso elemental que envía datos o información de control fuera de los límites de la aplicación.
    - Objetivo: presentar información a través de la recuperación de datos o información de control de un ILF o ELF
    - No afecta al sistema ni contiene datos derivados. Sólo recupera datos almacenados y se los proporciona al usuario
    - Ejemplo
      - Consulta de datos de un empleado.



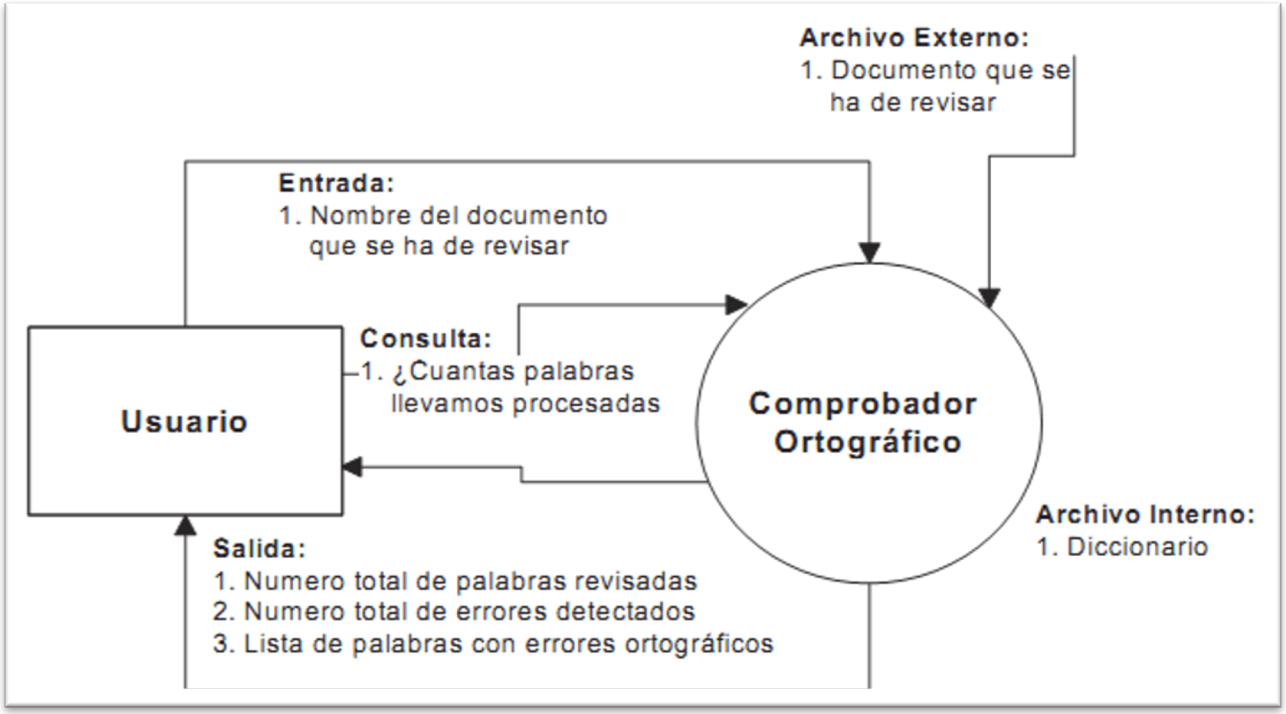
Elementos Funcionales
Resumen

- Entradas externas (entradas)
  - Cualquier entrada (pantalla, formulario, cuadro de diálogo, control o mensaje) que tenga un formato único o un solo procesamiento, a través de la cual el usuario u otro programa puede añadir, borrar o cambiar datos.
- Salidas externas (salidas)
  - Cualquier salida (pantalla, informe, gráfico, mensaje) que tenga un formato diferente o requiera un procesamiento diferente a otros tipos de salida, generada para el usuario u otro programa.
- Consultas externas (consultas)
  - Combinaciones de entrada/salida en las que cada entrada genera una salid simple e inmediata.
- Archivos lógicos internos (archivos)
  - Principales grupos lógicos de datos de usuarios o de control que están controlados completamente por el programa (una tabla de un SGBDR).
- Archivos de interfaz externos (interfaces)
  - Cada uno de los grupos de datos lógicos o información de control que entra o sale del programa

- Proceso (Método Estándar de IFPUG)



- Calcular los Puntos Función no ajustados (PFNA)



- **Entradas**: 1 (el nombre del archivo que ha de revisarse),
- **Salidas**: 3 (el número total de palabras revisadas, el número total de errores y una lista de las palabras erróneamente escritas),
- **Consultas**: 1 (el usuario puede obtener interactivamente el número de palabras procesadas hasta el momento),
- **Archivos**: 1 (el diccionario), y
- **Interfaces**: 1 (el documento a inspeccionar).

El número total de funciones de usuario es 1+3+1+1+1=7.

- **Evaluar la complejidad**
  - Determinar el nivel de complejidad (baja, media, alta) de cada función de usuario.
  - Para ello se tienen en cuenta el número de tipos de elementos de datos y el número de tipos de archivos (o de elementos de tipo registro) referenciados.
  - Aplicar pesos de complejidad, según el nivel de complejidad.

$$PF = \sum_{i=1}^{n} Ei * Ci$$

La suma ponderada de todas estas cantidades equivale al nº de PFNA

Técnicas de estimación
Puntos Función

Component	RET's	FTR's	DET's
External Inputs (EI)		✓	✓
External Outputs (EO)		✓	✓
External Inquiries (EQ)		✓	✓
External Interface Files (EIF)	✓		✓
Internal Logical Files (ILF)	✓		✓

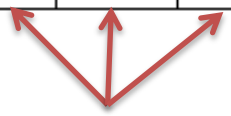
- **Evaluar la complejidad**
  - RET (Record Element Type)
    - Un subgrupo de datos de un fichero lógico interno o externo, reconocible por el usuario
  - DET (Data Element Type)
    - Un campo único de datos de un fichero lógico interno o externo, reconocible por el usuario
      - Entradas externas: campos de entrada de datos, mensajes de error, valores calculados, botones
      - Salidas externas: campos de datos en un informe, mensajes de error, valores calculados y encabezados de columna leídos de un ILF
      - Consultas externas: entrada → campo de búsqueda; salida → campo de datos recuperado
  - FTR (File Type Referenced)
    - Un fichero lógico interno leído o mantenido por una transacción ó un fichero externo de interfaz leído por una transacción

- **Evaluar la complejidad**
  - Criterios de IFPUG para evaluar la complejidad de elementos de cálculo en los puntos de función

Para ficheros lógicos internos y externos				**Para salidas y consultas**				**Para entrada**			
Registros elementales	Datos elementales			Tipos de ficheros	Datos elementales			Tipos de ficheros	Datos elementales		
	1-19	20-50	>51		1-5	6-19	>20		1-4	5-15	>16
1	Baja	Baja	Media	0-1	Baja	Baja	Media	0-1	Baja	Baja	Media
2-5	Baja	Media	Alta	2-3	Baja	Media	Alta	2-3	Baja	Media	Alta
>6	Media	Alta	Alta	>4	Media	Alta	Alta	>3	Media	Alta	Alta

- Evaluar la complejidad

Parámetros	Estimación			Complejidad			$PF_{sinAjuste}$
	Opt	*Prob*	*Pes*	*simple*	*media*	*alta*	Total
Nº de entradas de usuario				x 3	x 4	x 6	
Nº de salidas de usuario				x 4	x 5	x 7	
Nº de peticiones de usuario				x 3	x 4	x 6	
Grupos lógicos de datos internos				x 7	x 10	x 15	
Grupos lógicos de datos externos				x 5	x 7	x 10	
							Σ



Puede convenir realizar varias estimaciones:
Optimista, **Prob**able y **Pes**imista para considerar la media

- Ejemplo
  1. La aplicación debe almacenar y mantener empleados con los siguientes datos, nombre, número, dirección, fecha de nacimiento, oficina y la fecha en la que los datos del empleado fueron modificados por última vez.
  2. La aplicación debe proporcionar un medio para añadir nuevos empleados, actualizar su información, eliminar y combinar registros de empleados duplicados.
  3. La aplicación debe proporcionar un informe semanal con la información (nombre y número) de todos los empleados cuya información haya cambiado en los últimos siete días.
  4. La aplicación debe permitir consultar para una oficina determinada cuáles son todos sus empleados asignados.

SIGUE

- Ejemplo
  5. La aplicación debe proporcionar la posibilidad de ver todos los datos de un empleado.
  6. Los datos de seguridad del usuario (Identificación del usuario, Clave) son referenciados desde la aplicación de seguridad para validar el acceso del usuario a la aplicación.
  7. Se deberán utilizar algoritmos complejos de encriptación para la fecha de nacimiento, de manera que no pueda ser directamente leída a partir de la información almacenada de un empleado.
  8. La aplicación debe utilizar lenguajes de programación que sean compatibles con el diseño de sistemas abiertos y bases de datos Oracle.

- Ejemplo
  - Los requisitos 1-6 son los requisitos funcionales de los usuarios, mientras que los requisitos 7-8 se refieren a requisitos no funcionales.
  - Por ello, para calcular los PF de la aplicación se utilizarán los requisitos 1-6.

- Ejemplo
  - 1. Identificar las funciones de datos
    - 1.1 Determinar ILF
      - 1 ILF por los datos de los empleados que son datos persistentes mantenidos por la aplicación.
    - 1.2 Determinar ELF
      - 1 ELF por la entidad seguridad mantenida externamente.
  - 2. Identificar funciones de transacciones

- Ejemplo
  - 2. Identificar funciones de transacciones
    - 2.1. Determinar los procesos elementales
      - P1. Agregar un empleado
      - P2. Actualizar un empleado
      - P3. Eliminar un empleado
      - P4. Combinar registros duplicados
      - P5. Obtener el informe semanal
      - P6. Consultar empleados asignados a una oficina
      - P7. Control de la seguridad de acceso
      - P8. Mostrar los datos de un empleado (se debe utilizar un algoritmo de des-encriptación de la fecha de nacimiento)

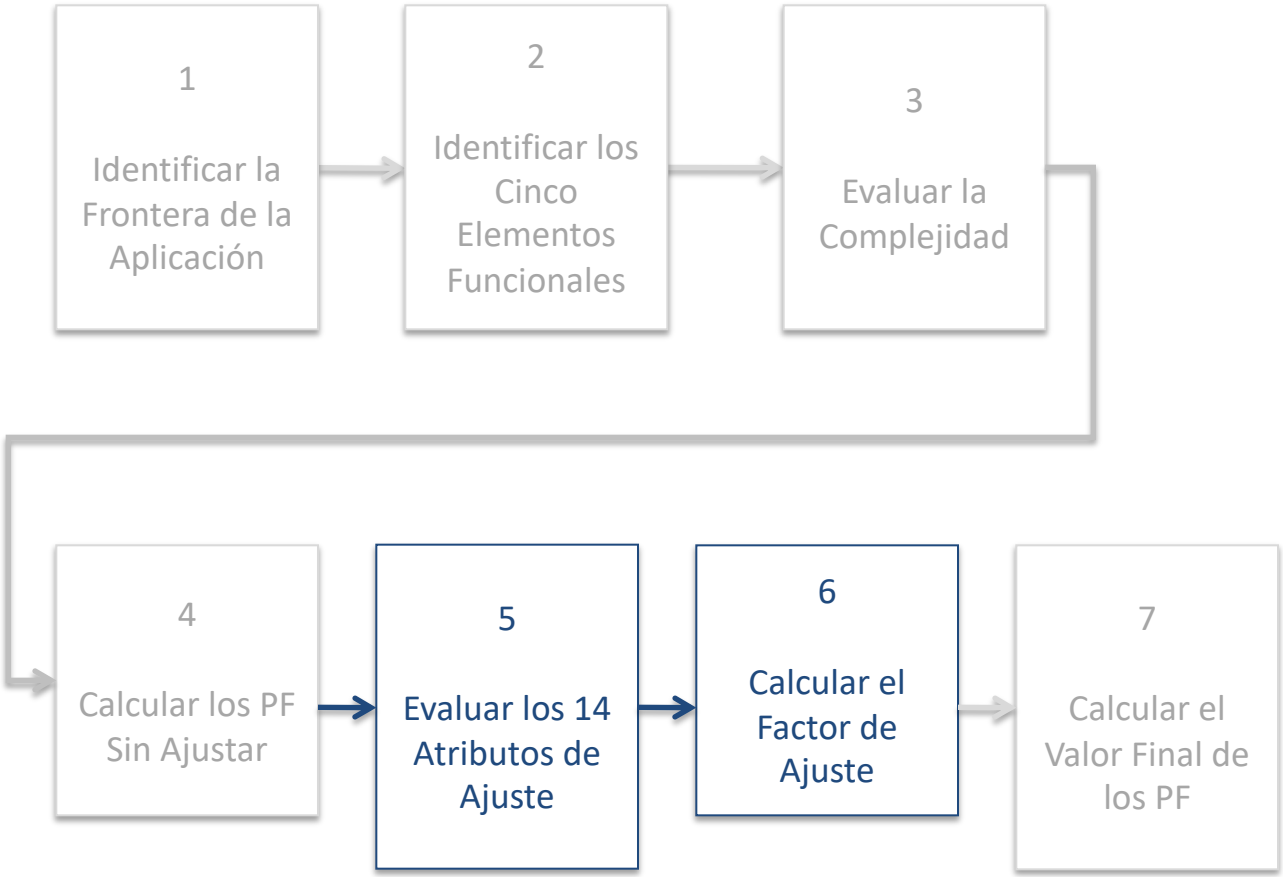
- **Ejemplo**
  - 2.2. Determinar los EI
    - Los datos se reciben desde fuera de la aplicación y actualizan ILFs
  - 2.3. Determinar los EQ
    - Sólo recuperan datos y no mantienen ningún ILF.
  - 2.4. Determinar los EO
    - Aplicar un algoritmo de cálculo.

- P1. Agregar un empleado
- P2. Actualizar un empleado
- P3. Eliminar un empleado
- P4. Combinar registros duplicados
- P5. Obtener el informe semanal
- P6. Consultar empleados asignados a una oficina
- P7. Control de la seguridad de acceso
- P8. Mostrar los datos de un empleado (se debe utilizar un algoritmo de des-encriptación de la fecha de nacimiento)

- Ejemplo

Elemento	Peso	Cantidad	Total = Cantidad * Peso
ILF	10	1	10
ELF	7	1	7
EI	4	4	16
EO	5	1	5
EQ	4	3	12
		Total PF	**50**
		-20%	**40**
		20%	**60**

- Proceso (Método Estándar de IFPUG)



- **Evaluar los 14 Atributos de Ajuste**
  - Ajustar el valor obtenido a las características del proyecto
    - El Factor de Ajuste (FA), se calcula a partir del grado de influencia de 14 Factores de Complejidad (FC) diferentes:

1. Comunicaciones de datos
2. Procesamiento distribuido
3. Objetivos de rendimiento
4. Configuración de uso intensivo
5. Tasas de transacción rápidas
6. Entrada de datos en línea
7. Amigabilidad en el diseño
8. Actualización de datos en línea
9. Procesamiento complejo
10. Reusabilidad
11. Facilidad de instalación
12. Facilidad operacional
13. Adaptabilidad
14. Versatilidad

    - A cada factor de complejidad se le asigna un peso entre 0 y 5 (aceptando decimales) en base al nivel de influencia que tiene sobre el software:

$$FA = 0{,}65 + 0.01 * SUMA(FC)$$
$$PF = PFSA * FA$$

0 => ninguna,
1 => muy poca,
2 => moderada,
3 => media,
4 => significativa, y
5 => esencial (mucha).

    - El factor FA puede oscilar entre 0,65 y 1,35, es decir, permite una variación máxima de ± 35% sobre el valor de los PFNA.

- **Estimación de tamaño: Puntos Función – EJEMPLO**
  - Podríamos usar directamente los PF (tamaño) para hacer una estimación del esfuerzo usando datos históricos

Parámetros	Estimación				Complejidad	$PF_{sinAjuste}$
	Opt	*Prob*	*Pes*	*Media*	*alta*	Total
Nº de entradas de usuario	8	11	17	12	x 3 (baja)	36
Nº de salidas de usuario	9	12	22	13	x 5(media)	65
Nº de peticiones de usuario	12	17	21	17	x 4(media)	68
Grupos lógicos de datos internos	3	4	5	4	x 10(media)	40
Grupos lógicos de datos externos	2	2	3	2	x 5(baja)	10
						Σ 219

Valores de entrada

$$PF_{ajustado} = PF_{sin\ ajustar}\ (0{,}65 + 0{,}01\ \Sigma\ F_i)$$

$$PF_{ajustado} = 219\ \underbrace{(0{,}65 + 0{,}01\ \Sigma\ F_i)}_{1{,}095} = 240 \qquad \Sigma\ F_i = 44{,}5$$

Ajustes a la naturaleza del proyecto

Productividad: 16PF/pm
Coste: 3.200 €/pm

Datos Históricos

$$\text{Coste/PF} = \frac{3.200\ €/pm}{16\ PF/pm} = 200\ €/PF$$

$$\text{Coste total} = 200\ €/PF\ 240\ PF = 48.000\ €$$

$$\text{Esfuerzo} = \frac{240\ LDC}{16\ PF/pm} = 15\ pm$$

Cálculo de Costes y Esfuerzos

Técnicas de estimación
Puntos Función

- **Estimación de tamaño**
  - Los PF pueden usarse para estimar LDC
    - LDC = AVC * PF
    - AVC es un factor que depende del lenguaje
    - 200-300 para Ensamblador a 2-40 para 4GL

Lenguaje	1PF →X LDC
4GL	40
Ada 83	71
Ada 95	49
APL	32
BASIC - compilado	91
BASIC -interpretado	128
BASIC ANSI/Quick/Turbo	64
C	128
C++	29
Clipper	19
Cobol ANSI 85	91
Delphi	1
Ensamblador	320
Ensamblador (Macro)	213
Forth	64
Fortran 77	105

Lenguaje	1PF→X LDC
FoxPro 2.5	34
Generador de Informes	80
Hoja de Cálculo	6
Java	53
Modula 2	80
Oracle	40
Oracle 2000	23
Paradox	36
Pascal	91
Pascal Turbo 5	49
Power Builder	16
Prolog	64
Visual Basic 3	32
Visual C++	34
Visual Cobol	20

Técnicas de estimación
Puntos Función

- Estimación de tamaño
  - Los PF pueden usarse para estimar LDC
    - LDC = AVC * PF
    - AVC es un factor que depende del lenguaje
    - 200-300 para Ensamblador a 2-40 para 4GL

http://www.qsm.com/resources/function-point-languages-table

Language	QSM SLOC/FP Data			
	Avg	Median	Low	High
ABAP (SAP)	18	18	16	20
Access	36	38	15	47
Ada	154	-	104	205
Advantage	38	38	38	38
APS	86	83	20	184
ASP	56	50	32	106
Assembler	209	203	91	320
C	148	107	22	704
C++	59	53	20	178
C#	58	59	51	66
Clipper	40	39	26	53
COBOL	80	78	8	400
ColdFusion	68	56	52	105
Cool:Gen/IEF	37	35	10	180
Culprit	51	-	-	-
Datastage	67	79	16	85
DBase III	-	-	-	-
DBase IV	52	-	-	-
Easytrieve+	33	34	25	41
Excel	47	46	31	63

Language	QSM SLOC/FP Data			
	Avg	Median	Low	High
Focus	45	45	40	49
FORTRAN	90	118	35	-
FoxPro	36	35	34	38
HTML	43	42	35	53
Ideal	66	52	34	203
Informix	42	31	24	57
J2EE	57	50	50	67
Java	55	53	9	214
JavaScript	54	55	45	63
JCL	96	59	58	173
JSP	59	-	-	-
KML	50	50	49	50
Lotus Notes	23	21	15	46
Maestro	30	30	30	30
Mantis	71	27	22	250
Mapper	69	70	58	81
Natural	51	53	34	60
.NET	60	60	60	60
Netron/CAP	296	323	105	399
Openroad	39	34	20	69

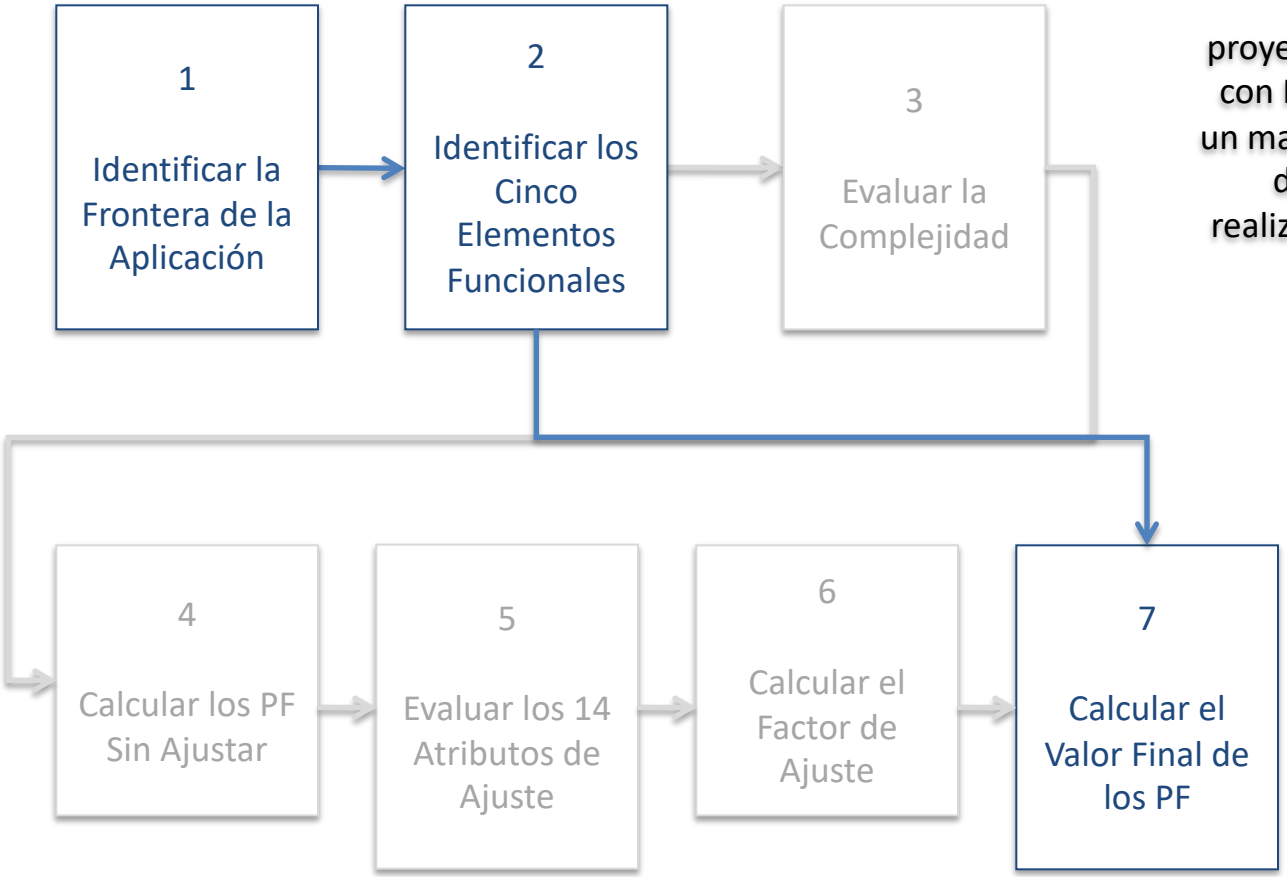
Estimación de tamaño: Puntos Función

COSMIC-FFP	FISMA 1.1	IFPUG 4.2	Mk II	NESMA
Identifica los procesos funcionales	Identifica los casos de uso de negocio	Identifica funciones de datos y funciones transaccionales	Identifica transacciones lógicas y tipos de entidad de datos	Identifica y corrige los límites del proyecto
Identifica y confecciona una lista de movimientos de datos de cada proceso funcional: • Entry • Exit • Read • Write	Identifica y confecciona una lista de servicios funcionales pertenecientes a cada grupo de servicios: • Navegación y servicios de consulta • Servicios interactivos de entrada • Servicios no interactivos de salida • Servicios de almacenamiento de datos • Servicios de interfaz hacia otras aplicaciones • Servicios de interfaz desde otras aplicaciones • Servicios de manipulación algorítmica	Identifica y realiza una lista de las funciones de datos pertenecientes a cada tipo de función de datos: • Fichero lógico interno • Fichero lógico externo Identifica y realiza una lista de funciones transaccionales pertenecientes a cada tipo de función: • Entrada externa • Salida externa • Consulta externa	Identifica y realiza una lista de tipos de datos elementales para cada transacción lógica: • Tipos de datos elementales de entrada • Tipos de entidad de datos referenciados • Tipos de datos elementales de salida	Identifica y confecciona una lista de funciones pertenecientes a cada tipo de función: • Fichero lógico interno • Fichero lógico externo • Entrada externa • Salida externa • Consultas externas

[Piattini, 2008]
ISO/IEC 14143-1

- **Simplificaciones de PF**
  - FP Lite (David Consulting Group, 2007)
    - Versión simplificada de FP, en la que se considera que todos los elementos tienen complejidad media
  - *Puntos Casos de Uso* (Karner, 1993)
    - Ganando popularidad, comienza con casos de usos y actores, en lugar de entradas, salidas, etc.
  - Early and Quick Function Points (E&QFP) (Roberto Meli, ESCOM 97)

- Proceso (Método FP Lite)



El 70% de los proyectos estimados con FP Lite están en un margen de ±20% de la estimación realizada con FPA de la IFPUG

- **Estimación de tamaño: Puntos Función**
  - Ventajas
    - Independientes del lenguaje
    - Pueden calcularse a partir de la especificación
  - Desventajas
    - Naturaleza subjetiva → Dependencia del estimador
      - » Lo que para ti es complejo para mi no lo es tanto
    - Predispuestos para sistemas de proceso de datos, con alta carga de I/O
      - » No se adapta bien a sistemas dirigidos por eventos

- **Estimación de tamaño: Puntos Objeto**
  - Puntos Objeto (desarrollos con 4GL)
    - Nº de pantallas independientes
      - Sencillas | Complejas | Muy Complejas → 1 | 2 | 3 pts.
    - Nº de informes
      - Sencillas | Complejos | Muy Complejos → 2 | 5 | 8 pts.
    - Nº de módulos de lenguaje imperativo necesario para complementar la BBDD (3GL para completar 4GL)
      - 10 puntos por módulo
    - Ventajas
      - Igual que PF, pueden estimarse a partir de una especificación de muy alto nivel, fases en las que es muy complejo estimar LDC

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
      - (…) COCOMO
  - Métodos de Estimación orientados a objetos
  - Buenas prácticas y lecciones aprendidas

Estimación de Esfuerzo: Modelos empíricos

- También conocidos como modelos algorítmicos de costes
- Se utilizan datos históricos y estimaciones del tamaño o PF sobre fórmulas empíricas
- Existen una gran cantidad de fórmulas empíricas y su aplicación depende del tipo de proyecto y del entorno de desarrollo. Los métodos empíricos de estimación siguen esta fórmula genérica:

$$\textbf{Esfuerzo} = A \times \text{Tamaño}^{B} \times M$$

- Donde
  - A = f(practicas organizacionales y tipo SW)
  - B = f(atributos producto y equipo)
  - M= f(practicas organizacionales, tipo SW, atributos producto y equipo)
- Cada método adapta esta fórmula, modificando principalmente el valor de las constantes.

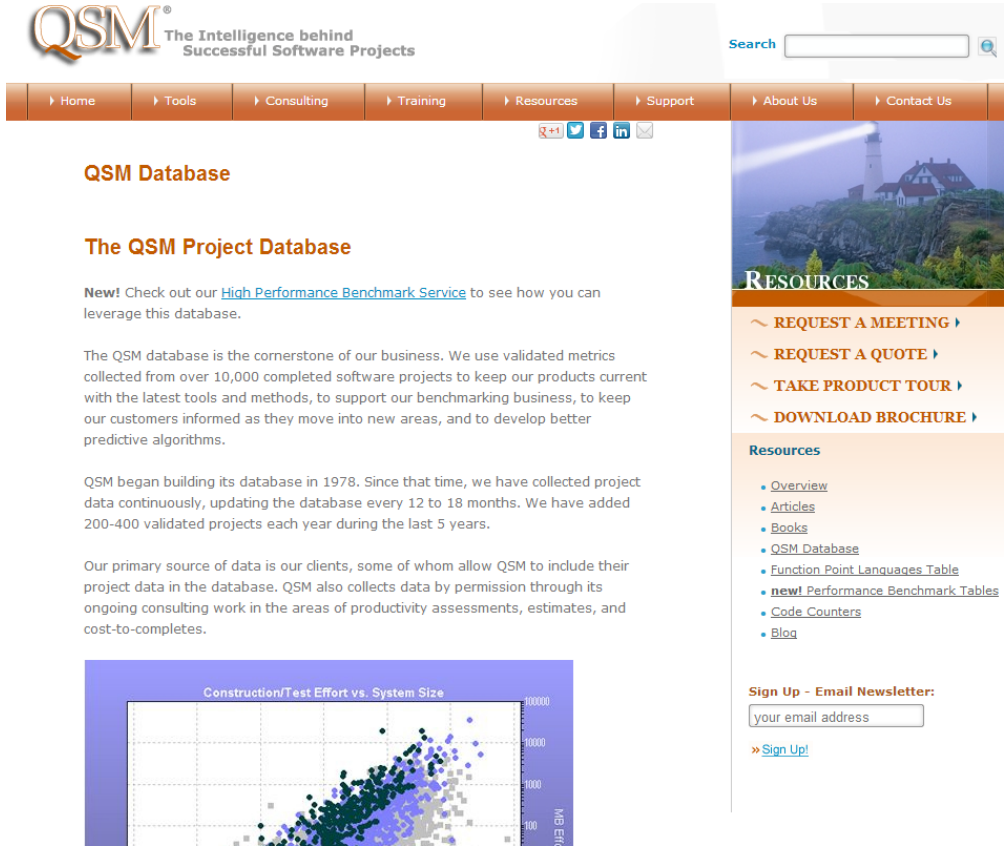
Relación Tamaño/Esfuerzo no es lineal

[Sommerville, 2005]

- **Estimación de Esfuerzo**
  - Modelos empíricos de estimación
    - Basados en LDC
      - E = 5,2 x (KLDC)0,91 — Modelo de Walston-Felix
      - E = 5,5 + 0,73 x (KLDC)0,16 — Modelo de Bailey-Basili
      - E = 3,2 x (KLDC) 0,5 — Modelo simple de Boehm
      - E = 5,288 x (KLDC) 1,047 — Modelo Doty para KLDC > 9
    - Basados en PF
      - E = -13,39 + 0,0545 PF — Modelo de Albretch y Gaffney
      - E = 60,62 x 7,728 x 10-8 x PF3 — Modelo de Kemerer
      - E = 585,7 + 15,12 PF — Modelo de Matson, Bamett y Mellichamp

Coge uno y adáptalo!!!
shu-ha-ri
守破離

- **SLIM (Software Lifecycle Management) – Putnam**
  - Estudian una Base de Datos (QSM):
    - 750 sistemas procedentes de la Air Force Electronic Systems Division, Rome Air Development Center y otros sistemas de procedencias diversas.



Putnam, Lawrence H. (1978). "A General Empirical Solution to the Macro Software Sizing and Estimating Problem". IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. SE-4, NO. 4, pp 345-361.

- SLIM (Software Lifecycle Management) – Putnam

Producto = PP * (Esfuerzo / B) $^{(1/3)}$ * Tiempo $^{(4/3)}$

  - Producto se mide en SLOC
  - Parámetro de productividad (PP): Se obtiene por calibración con datos históricos de proyectos, capacidad de desarrollo, se suele derivar de datos históricos aplicando la ecuación.
  - Esfuerzo en hombres-año: comprende diseño, codificación, test, documentación y supervisión.
  - B es un parámetro de habilidad: depende del tamaño del producto y se calcula sobre datos históricos
  - Tiempo (o Td) de desarrollo (diseño, codificación, test, documentación y supervisión) en años

- **Estimación de Esfuerzo:**
  - Método COCOMO (Modelo de Construcción de Coste)
    - Es el modelo de estimación más utilizado.
    - No es en realidad un modelo de estimación, sino una jerarquía de modelos, ya que se proponen distintos tipos de modelos para distintos tipos de situaciones (básico, intermedio y detallado)
    - La primera versión surge en 1981 y en 1995 surge la versión COCOMO II
    - Sus autores (Center for Software Engineering, University of Southern California) pretenden mejorar, ampliar y adaptar el modelo anterior a las nuevas formas en que se desarrolla el software:
      - Nuevas aproximaciones: desarrollo evolutivo, dirigido a riesgos, colaborativo.
      - Nuevos entornos: 4GL's, generadores de aplicaciones, orientación a objetos, ...
      - Nuevos paradigmas: reusabilidad, madurez, calidad total, …

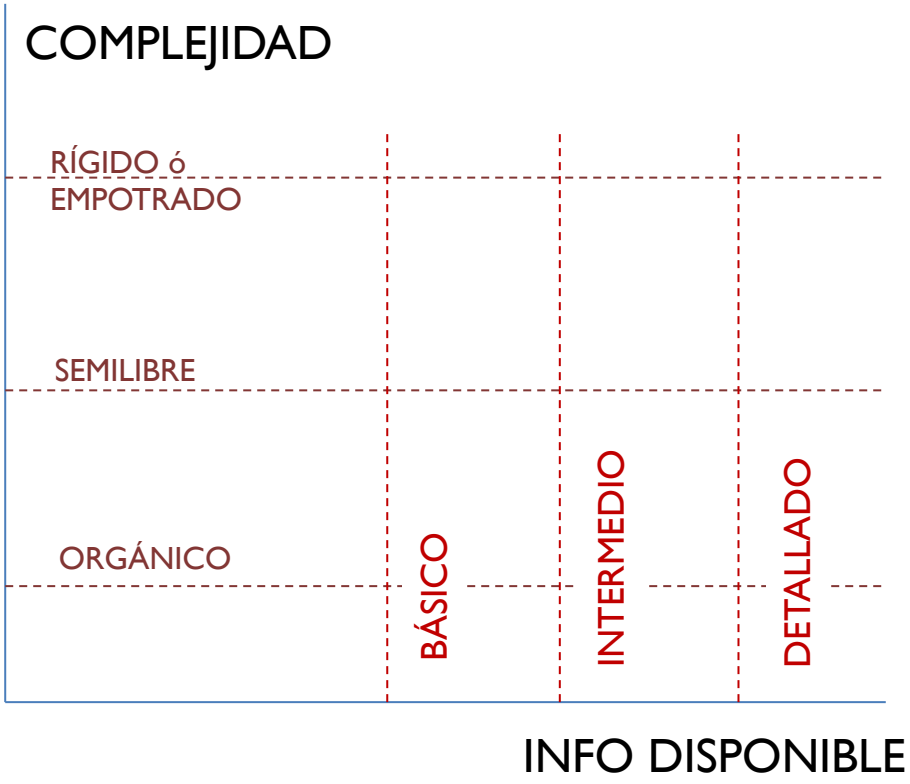
[Boehm, 1981]

- COCOMO
  - Carácterísticas
    - Moderadamente preciso
    - Se calcula el esfuerzo en función del tamaño del software (LDC), aplicando la ecuación básica

COCOMO básico:	**Esfuerzo = $a*(KLDC)^b$**
COCOMO II:	**Esfuerzo = $a*(KLDC)^b * m(x)$**

- El esfuerzo de desarrollo (ED) se mide en personas-mes (PM)
- a y b son los parámetros de ajuste según el tipo de desarrollo del proyecto (**modo**)
- Los coeficientes varían para los diferentes modos
- m(X) es un multiplicador que depende de 15 atributos
  - No se considera en COCOMO básico

Técnicas de Estimación: COCOMO



- **COCOMO: Tres modos de desarrollo de proyectos Software**
  - Modo orgánico (< 50 KLDC)
    - Desarrollo en un entorno estable, con poca innovación técnica, con pocas presiones de tiempo y tamaño relativamente pequeño (< 50 KLDC).
    - Áreas específicas y bien conocidas por el equipo.
    - El tamaño del software varía de unos pocos miles de líneas (tamaño pequeño) a unas decenas de miles de líneas (medio), mientras que en los otros dos modos el tamaño varía de pequeño a muy grandes (varios cientos de miles de líneas).
    - En este modo, al igual que en los otros, el coste se incrementa a medida que lo hace el tamaño , y que el tiempo de desarrollo se alarga.

- **COCOMO: Tres modos de desarrollo de proyectos software**
  - Modo semilibre o semiencajado (< 300 KLDC)
    - Corresponde a un esquema intermedio entre el orgánico y el rígido
    - El grupo de desarrollo se sitúa en niveles (incluye personas experimentadas y no experimentadas)
    - Suelen ser sistemas con interfaces con otros sistemas y un tamaño menor a 300 KLDC.
  - Modo rígido o empotrado
    - Desarrollo de software con requisitos muy restrictivos, con gran volatilidad de requisitos, complejo, en un entorno con gran innovación técnica.
    - Participan muchas personas y se requiere un alto grado de fiabilidad.
    - El problema a resolver es único y es difícil basarse en la experiencia, puesto que puede no haberla.

- COCOMO
  - Jerarquía de modelos de estimación según el nivel de detalle empleado en su utilización:
    - COCOMO básico
    - COCOMO intermedio
    - COCOMO detallado

- COCOMO
  - COCOMO BÁSICO

$ED = a*(KLDC)^{b}$

  - Para estimaciones al inicio del proyecto cuando no se dispone de detalles: por ejemplo, al empezar a negociar el contrato
  - Moderadamente preciso
  - Se calcula el esfuerzo en función del tamaño del software (LDC), aplicando la ecuación básica.

Modo de desarrollo	ED: Esfuerzo de Desarrollo (nominal) Personas-mes	TD: Tiempo de desarrollo (nominal)
Orgánico	$ED = 2.4 \times KLDC^{1.05}$ PM	$TD = 2.5 \times PM^{0.38}$
Semi-libre	$ED = 3.0 \times KLDC^{1.12}$ PM	$TD = 2.5 \times PM^{0.35}$
Empotrado	$ED = 3.6 \times KLDC^{1.20}$ PM	$TD = 2.5 \times PM^{0.32}$

A medida que aumenta la complejidad del proyecto, aumenta el esfuerzo necesario

- COCOMO
  - COCOMO INTERMEDIO

$$ED = a*(KLDC)^{b}*m(x)$$

    - Cuando tenemos identificados los principales componentes del sistema.
      - Por ejemplo, se dispone de una especificación de requisitos más o menos terminada.
    - Se estima el coste de dichos componentes:
      - Aplicando la ecuación básica para obtener el esfuerzo o el tiempo de desarrollo. Este desarrollo se denomina nominal, ya que no está adaptado a las características del entorno de desarrollo.
      - Este esfuerzo nominal se ajusta incorporando la influencia de 15 factores de coste.

- COCOMO
  - Ecuaciones para el cálculo del esfuerzo y el tiempo de desarrollo nominal para COCOMO INTERMEDIO

Modo de desarrollo	ED:Esfuerzo de Desarrollo (nominal) Personas-mes	TD: Tiempo de desarrollo (**nominal**)
Orgánico	$ED = 3.2 \times KLDC^{1.05}$ PM	$TD = 2.5 \times PM^{0.38}$
Semi-libre	$ED = 3.0 \times KLDC^{1.12}$PM	$TD = 2.5 \times PM^{0.35}$
Empotrado	$ED = 2.8 \times KLDC^{1.20}$PM	$TD = 2.5 \times PM^{0.32}$

  - Este esfuerzo nominal se ajusta incorporando la influencia de los 15 factores de coste mencionados

$$ED = a*(KLDC)^{b}*m(x)$$
$$m(x) = FC1*FC2*\ldots$$

COCOMO

– Factores de coste en
  - Software
  - Hardware
  - Personal
  - Proyecto

$ED = a*(KLDC)^{b} * m(x)$

m(X) = producto FCs
•En ausencia de información, se considera FC nominal → FC = 1
•Por tanto no afecta al cálculo del esfuerzo ajustado

Atributos	Valor					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extra alto
Atributos de software						
Fiabilidad	0,75	0,88	1,00	1,15	1,40	
Tamaño de Base de datos		0,94	1,00	1,08	1,16	
Complejidad	0,70	0,85	1,00	1,15	1,30	1,65
Atributos de hardware						
Restricciones de tiempo de ejecución			1,00	1,11	1,30	1,66
Restricciones de memoria virtual			1,00	1,06	1,21	1,56
Volatilidad de la máquina virtual		0,87	1,00	1,15	1,30	
Tiempo de respuesta		0,87	1,00	1,07	1,15	
Atributos de personal						
Capacidad de análisis	1,46	1,19	1,00	0,86	0,71	
Experiencia en la aplicación	1,29	1,13	1,00	0,91	0,82	
Calidad de los programadores	1,42	1,17	1,00	0,86	0,70	
Experiencia en la máquina virtual	1,21	1,10	1,00	0,90		
Experiencia en el lenguaje	1,14	1,07	1,00	0,95		
Atributos del proyecto						
Técnicas actualizadas de programación	1,24	1,10	1,00	0,91	0,82	
Utilización de herramientas de software	1,24	1,10	1,00	0,91	0,83	
Restricciones de tiempo de desarrollo	1,22	1,08	1,00	1,04	1,10	

- COCOMO
  - COCOMO detallado
    - Cuando están identificados los componentes individuales del sistema.
      - Por ejemplo, cuando se dispone de una especificación de requisitos totalmente acabada o cuando el diseño general está bien definido
    - Proporciona tablas para ajustar el esfuerzo y el tiempo en las distintas fases de desarrollo del proyecto.
    - Las fórmulas de estimación también varían en función de la granularidad del componente del software a estimar: módulo, subsistema o sistema. La cuantificación se realiza al nivel al que es más susceptible la variación.
    - También proporciona tablas para poder refinar el ajuste de acuerdo al entorno de desarrollo.

- Resumen usando COCOMO básico:

	Orgánico	Semiempotrado	Empotrado
Esfuerzo de desarrollo	$ED=2,4(KLDC)^{1,05}$ pm	$ED=3,0(KLDC)^{1,12}$ pm	$ED=3,6(KLDC)^{1,20}$ pm
Tiempo de desarrollo	$TD=2,5(ED)^{0,38}$ m	$TD=2,5(ED)^{0,35}$ m	$TD=2,5(ED)^{0,32}$ m
Productividad	PR = LDC/ED		
Nº medio de personas a tiempo completo	FSP (Full-Time equivalent Software Personel) PE= ED/ TD p		
Esfuerzo de Mantenimiento	TCA (Tráfico de cambio anual): porción de instrucciones fuente que sufren algún cambio durante un año, bien sea por adición o por modificación. Se calcula como LDC que varían/LDC totales EM = TCA x ED Y por tanto el valor medio del número de personas a tiempo completo, dedicadas a mantenimiento durante 12 meses sería: PEM= EM/12		

pm → Personas/Mes m → Mes p → Personas

- COCOMO
  - Ejemplo
    - Se trata de estimar el esfuerzo de desarrollo de un sistema de comunicaciones de 30 KDLC, de alta complejidad. Afortunadamente podremos emplear personal (programadores) de muy alta cualificación con una gran experiencia específica en este tipo de software.
    - Sabiendo que el coste del salario mensual de cada persona es de 1.000 euros/mes ¿Sería más rentable emplear a personas de nivel medio cuyo salario es 850 euros?

- COCOMO
  - Ejemplo (SOLUCIÓN)
    - KLDC = 30 KLDC < 50 KLDC → Modo orgánico
    - Esfuerzo nominal = $3,2 \times (30)^{1,05} = 113,79$ personas-mes.
  - Ajustando el esfuerzo nominal
    - ED = Esfuerzo = 113,79 × 1,15 (complejidad) × 0,70 (personal) × 0,91 (experiencia) = 83,35 pm
    - Coste = 83,35 pm x 1000 €/mes = 83.350 euros
    - TD = Tiempo = $2,5 \times PM^{0,38} = 2,5 \times 83,350.38 = 13,42$ meses
    - PE = número medio de personas = ED/TD = 83,35/13,42 = 6,2 personas

- COCOMO
  - Ejemplo (SOLUCIÓN)
    - Coste solución de partida = 83.300 euros
    - Esfuerzo ajustado si empleamos personal de nivel medio
      - El esfuerzo nominal no varía: 113,79 personas-mes
      - ED = 113,79 × 1,15 (complejidad) × 1 (personal) × 0,91 (experienc.) = 119,08 pm
      - Coste = 119,08 pm x 850 €/mes = 101.218 euros

Sabiendo que el coste del salario mensual de cada persona es de 1.000 €/mes ¿Sería más rentable emplear a personas de nivel medio cuyo salario es 850 €?

A pesar del ahorro en salarios, el coste del proyecto contratando personal de nivel medio acabaría siendo máyor que el coste asociado a emplear personal altamente cualificado

101.218 > 83.300

- # COCOMO II

[Boehm et al., 2000]

  - En 1995 se publicó la versión COCOMO II
  - Objetivos:
    - Desarrollar un modelo de estimación de costes y tiempos en consonancia con las prácticas actuales de ciclo de vida del software.
    - Construir una base de datos y una herramienta de costes del software que incluya capacidades para la mejora continua del modelo.
    - Proveer un marco analítico cuantitativo, y un conjunto de herramientas y técnicas para evaluar los efectos de las mejoras en la tecnología software sobre los costes y tiempos del ciclo de vida del software.

- COCOMO II
  - Se proponen tres modelos distintos en función del tipo de desarrollo y de la fase de desarrollo en la que se desea estimar
    - Los modelos COCOMO II requieren información de tamaño
    - Cada modelo utiliza una magnitud diferente

Modelo	Se basa en	Recomendado para
Application Composition Model (CAM)	Nº Puntos Objeto (PO)	Desarrollo basado en prototipos y/o componentes
Early Design Model (EDM)	Nº Puntos Función (PF)	Estimación inicial basada en requisitos y primeros diseños
Post-Architecture Model (PAM)	Nº Líneas de Código (LDC)	Diseño arquitectónico (estructura de subsistemas)

- COCOMO II
  - Mantiene una jerarquía de modelos, según la etapa de desarrollo en la que se realice la estimación:
    - **Modelo de Composición de Aplicaciones (CAM).** Se utiliza en la etapa inicial de los proyectos de desarrollo. Se utiliza fundamentalmente para la estimación de actividades de prototipado, propias de fases muy incipientes del ciclo de desarrollo.
    - **Modelo de Diseño Temprano (EDM).** Se utiliza generalmente en las fases iniciales de la especificación, después de obtener los requisitos y diseñar la arquitectura básica del sistema, por ejemplo para explorar las diferentes alternativas arquitectónicas o tecnológicas.
    - **Modelo posterior al Diseño Arquitectónico (PAM).** Está orientado a las etapas de desarrollo y mantenimiento del producto software, cuando ya se dispone de una especificación completa y detallada.

- # COCOMO II: Modelo de Composición de Aplicaciones –CAM- (Application Composition Model)

  - Se usa para estimación de esfuerzo iniciales, para desarrollo basado en componentes (reutilización extensiva) y prototipado.
  - Para la estimación se basa en Puntos Objeto (PO)
    - Recordad: en función de pantallas, informes y componentes (3GL), requeridos para construir la aplicación.

- # COCOMO II: Modelo de Composición de Aplicaciones (CAM)
  - Para la estimación se basa en **Puntos Objeto** (PO):
    1. Cada instancia de objeto (por ejemplo, una pantalla o informe) se clasifica en base a tres niveles de complejidad: simple, medio y difícil.
    2. Una vez determinada la complejidad se pondera en base a la siguiente tabla:

Tipo de objeto	Peso de la complejidad		
	Simple	Medio	Difícil
Pantalla	1	2	3
Informe	2	5	8
Componente 3GL			10

    3. El número de puntos de objeto se calcula como la suma del total de puntos de objeto multiplicado por su factor de ponderación.

$$NPO = \sum(PO \times FP)$$

PO → Cada instancia de PO
FP → Factor de Ajuste

- # COCOMO II: Modelo de Composición de Aplicaciones (CAM)

4. Se corrige el número de puntos de objetos en función de la reutilización de componentes:

NPO = PO x ((100-%reutilización)/100)

5. Se añade a la ecuación la tasa de productividad (PROD = NPO/PM):

PM = NPO/PROD

PM → esfuerzo (personas/mes)
NPO → numero de puntos de objeto
PROD → productividad

Expericiencia y capacidad de los desarrolladores y Madurez y capacidad de las Herramientas CASE	Muy Baja	Baja	Media	Alta	Muy Alta
PROD (NPO/mes)	4	7	13	25	50

- COCOMO II: Modelo de diseño inicial – DM– (Early Design Model):
  - Usado en las etapas iniciales: después de acordar los requisitos y con un diseño sólo inicial
  - Se pretende obtener una estimación aproximada sin demasiado esfuerzo
    - Para ello, asumimos simplificaciones: p.e. esfuerzo de reutilizar = 0
  - La estimación de tamaño se basa en PFNA → KLDC (conversión con tablas)
  - Utiliza 7 multiplicadores de esfuerzo (cost drivers) que afectan multiplicativamente al coste del proyecto: valores de [1 – 6]

$$PM = A \times KLDC^{B} \times M$$

A → 2,94 (basado en la experiencia)
M → RCPX x RUSE x PDIF x PERS x PREX x FCIL x SCED

– RCPX: Fiabilidad y complejidad del producto.
– RUSE: Reutilización requerida.
– PDIF: Dificultad de la plataforma.
– PERS: Capacidad del personal.
– PREX: Experiencia del personal.
– FCIL: Medios (facilities).
– SCED: Calendario.

  - B se calcula en función de 5 factores que afectan exponencialmente al coste del proyecto.

- # COCOMO II: Modelo posterior al diseño de la arquitectura – PAM (Post-Architecture Model):
  - Similar a EDM pero,

$$PM = A \times KLDC^{B} \times M$$

    - Incorpora17 multiplicadores de esfuerzo (cost drivers), en vez de los 7 de EDM

Factores del **Producto**: equivalentes a RCPX -los tres primeros- y RUSE -el último- en el modelo EDM.
– RELY: Fiabilidad del producto requerida.
– DATA: Tamaño de la base de datos.
– CPLX: Complejidad del producto.
– DOCU: Adecuación de la documentación a las necesidades del ciclo de vida.
– RUSE: Reutilización requerida.

Factores del **Personal**: equivalentes a PERS -los tres primeros- y PREX -los tres últimos- en el modelo EDM.
– ACAP: Capacidad de los analistas.
– PCAP: Capacidad del programador.
– PCON: Continuidad del personal.
– AEXP: Experiencia en aplicaciones.
– PEXP: Experiencia en la plataforma.
– LTEX: Experiencia con el lenguaje y las herramientas

Factores del **Proyecto**: equivalentes a FCIL -los dos primeros- y SCED -el último- en el modelo EDM.
– TOOL: Uso de herramientas software.
– SITE: Desarrollo en varios sitios.
– SCED: Calendario de desarrollo requerido.

Factores de la **Plataforma**: equivalentes a PDIF en el modelo EDM.
– TIME: Limitaciones en el tiempo de ejecución.
– STOR: Limitaciones en el almacenamiento principal.
– PVOL: Volatilidad de la plataforma.

- # COCOMO II

$$PM = A \times Tamaño^{B} \times M$$

  - Cálculo del Factor de Escala (B) en EDM y PAM:
    - Considera las diversas economías de escala, positivas o negativas, existentes en proyectos software.
    - Su valor depende de 5 factores de escala, asignando a cada uno un peso de 0 (muy alto) a 5 (muy bajo)..

$$B = 0{,}91 + 0{,}01 \times \sum W_i \ (i=1..5)$$

W_i

–PREC: Ausencia de Precedentes (Precedentedness).
–FLEX: Flexibilidad del desarrollo.
–RESL: Resolución Arquitectura/Riesgos (mide una combinación del uso de la gestión de riesgos y de la minuciosidad al diseñar la arquitectura del sistema).
–TEAM: Cohesión del equipo de personas participantes.
–PMAT: Madurez del proceso (basado en utilizar el modelo CMM - Capability Maturity Model).

- COCOMO II
  - Subjetividad!!!

Factor de Escala	1.17	
Tamaño del sistema	128, 000 LDC	
Estimación COCOMO sin multiplicadores de esfuerzo	**730 personas/mes**	E_i
Reliability	Muy Alta → 1.39	
Complexity	Muy Alta → 1.3	
Memory constraint	Alta →1.21	
Tool use	Baja → 1.12	
Schedule	Acelerada → 1.29	
Estimación COCOMO ajustada al alza	**2306 personas/mes**	E_i x 3
Reliability	Muy Baja → 0.75	
Complexity	Muy Baja → 0.75	
Memory constraint	Ninguna → 1	
Tool use	Muy Alta → 0.72	
Schedule	Normal → 1	
Estimación COCOMO ajustada a la baja	**295 personas/mes**	$E_i/3$

- Críticas a los modelos algorítmicos de Estimación
  - Los que dependen del NLDC tienen el inconveniente de que hay que calcular de alguna manera este parámetro.
    - La pura adivinación o deducción por expertos de este valor puede ser tan complicada como el propio cálculo del coste del proyecto.
    - Es complicado estimar el tamaño del SW en etapas tempranas donde sólo está disponible la especificación
  - Los factores de coste son difíciles de cuantificar y se consideran independientes aunque no necesariamente lo son.
    - Aunque hay guías para aplicar de manera uniforme y consistente los modelos, existe un alto grado de subjetividad en la estimación.
  - Los modelos surgen del análisis estadístico de datos de proyectos. El problema es que la cantidad y representatividad de proyectos no es tan amplia como sería deseable.
  - Los modelos pierden bastante precisión al utilizarse en entornos distintos de aquéllos en los que se crearon.
  - Tienen un cierto margen de error que, en algunos casos, es aceptable aunque, lamentablemente, en otros llega a cifras muy altas.

[Sommerville, 2005]

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de estimación orientados a objetos
    - Puntos Función Orientados a Objetos (OOFP)
    - Puntos Casos de Uso (UCP)
    - Puntos Clase
  - Buenas prácticas y lecciones aprendidas

- **Puntos Función Orientados a Objetos**
  - Mapeo de los conceptos contemplados en Punto-Función (ficheros lógicos y transacciones) a OO (clases y métodos)
  - Recomendaciones sobre cómo gestionar particularidades, como la herencia o la agregación
  - Trabaja fundamentalmente con el modelo de clases
    - Representación estática de clases y objetos
    - El primero que se elabora → permite realizar estimaciones tempranas

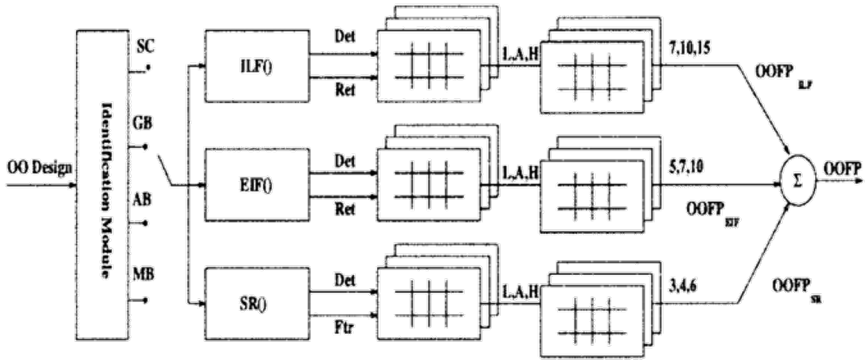
[Antoniol et al., 1999]

- **Fichero lógico - Clase**
  - Encapsulan conjunto de datos
- **Transacciones – Servicio**
  - *Service Requests*

Function point concept	Object point concept
Internal logical files	Internal class
External interface files	External class
External inquires	Services (methods)
External outputs	
External inputs	

OOFP

- **Proceso**
  - Identificar ficheros lógicos
  - Asignar complejidades a LF y SR
  - Convertir esas complejidades a valores numéricos
  - Si un LF es reutilizado escalar su valor (<=1)
  - Sumar todos los valores



OOFPs can be calculated as:

$$OOFP = OOFP_{ILF} + OOFP_{EIF} + OOFP_{SR}$$

where:

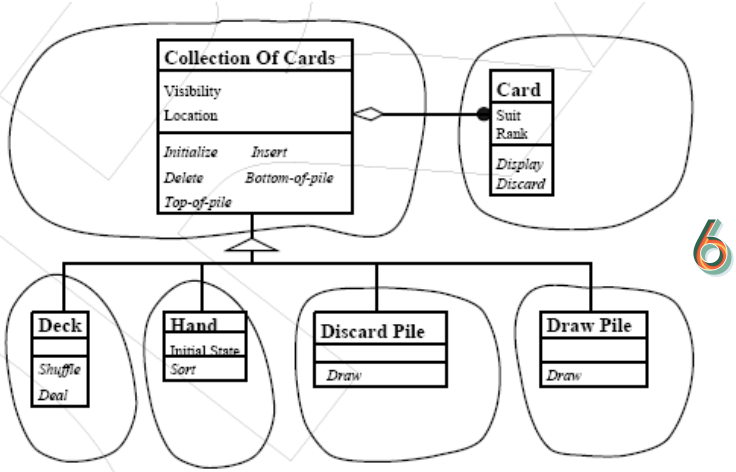
$$OOFP_{ILF} = \sum_{o \in A} W_{ILF}(DET_o, RET_o)$$

$$OOFP_{EIF} = \sum_{o \notin A} W_{ELF}(DET_o, RET_o)$$

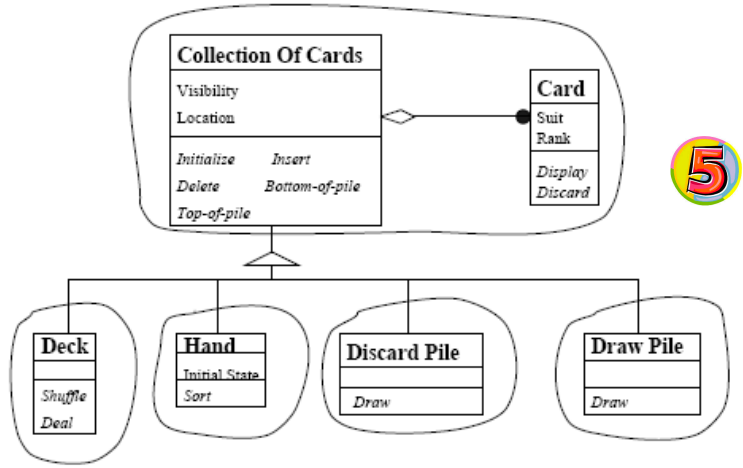
$$OOFP_{SR} = \sum_{o \in A} W_{SR}(DET_o, FTR_o)$$

- **Identificar ficheros lógicos (LF)**
  - Classes are mapped to LF
  - Aggregation and Inheritance is encountered (to group classes as unique LF)
    - Mainly a concern of implementation
  - At analysis phase
    - Each class is a LF
    - Scale factor =1 (Origin of class does not matter)
  - At design phase
    - Scale factor <1, reuse makes classes easier to develop
    - For designer, each class is LF
    - For user perspective, it is complicated

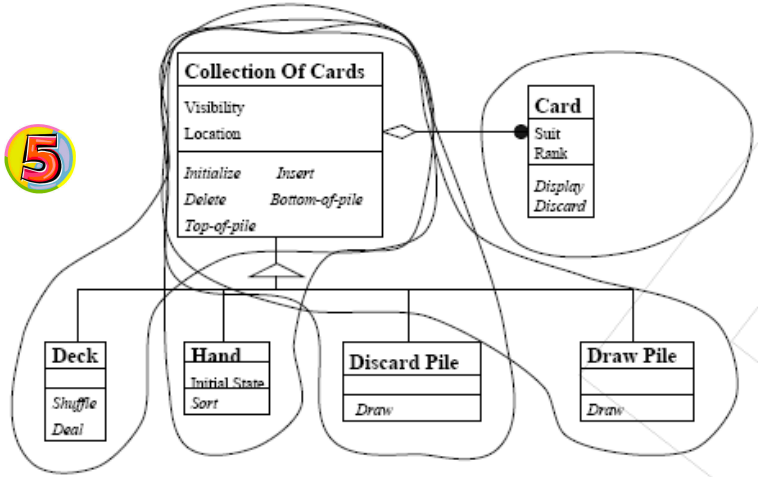
- **Ways to Identify LF**
  - Simple LF
    - Single Class is a LF
  - Composite LF
    - Aggregation
    - Generalization/Specialization
    - Mixed (combine aggregation and generalization)



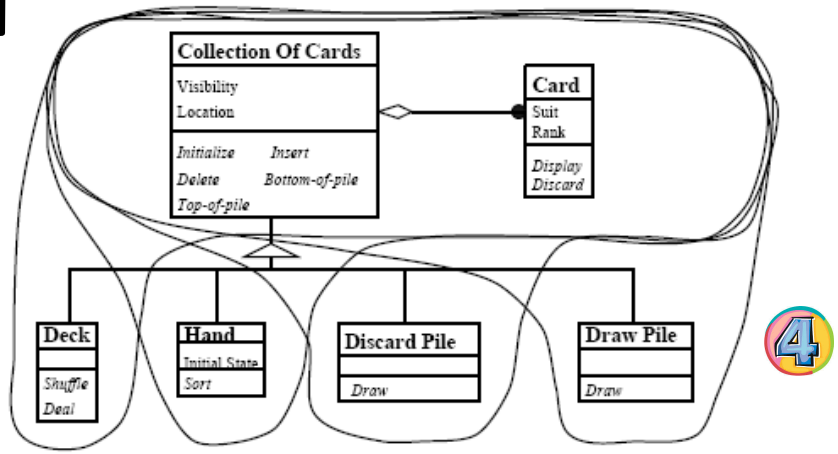
Single



Aggregation



Generalization



Mix

- **Calcution of D(ata)E(element)T(ypes) and R(ecord)ETs**
  - One RET for each ILF/ELF
  - Simple LF
    - Simple attributes such as integer and strings counted as DET
    - Associations are counted as DET or RET according to cardinality
      - Single valued association is DET
      - Multiple valued association is RET

- **Calcution of D(ata)E(element)T(ypes) and R(ecord)ETs**
  - Composite LF
    - DETs and RETs for each class are counted as in simple LF, except for aggregation
    - Aggregations act as subgroups in composite LF
      - One RET is counted for aggregations
      - It is assigned to the container class

OOFP

- For each OOFP , weighted vector table for ILF and ELF in IFPUG (international function point user group)

Type of Component	Complexity of Components			
	Low	Average	High	Total
External Inputs	___ x 3 = ___	___ x 4 = ___	___ x 6 = ___	
External Outputs	___ x 4 = ___	___ x 5 = ___	___ x 7 = ___	
External Inquiries	___ x 3 = ___	___ x 4 = ___	___ x 6 = ___	
Internal Logical Files	___ x 7 = ___	___ x 10 = ___	___ x 15 = ___	
External Interface Files	___ x 5 = ___	___ x 7 = ___	___ x 10 = ___	
		Total Number of Unadjusted Function Points		___
		Multiplied Value Adjustment Factor		___
		Total Adjusted Function Points		___

OOFP

- **Service Requests**
  - Concrete methods are only counted once (only coded once)
  - Abstract methods are not counted
    - Simple Items: (analogy to DET)
      - Simple data items referenced as a argument
      - Simple global variables referenced by the method
    - Complex Items: (analogy to FTR)
      - Complex arguments, objects and complex global variables references by the method
- **For each OOFP(SR), weighted vector table for EI, EQ in IFPUG**

- **SR:12 average methods**
  - 12 × 4 = 48 OOFPs
- **LF**
  - Card
    - 3 DET = 2 atts + 1 N-1 association
    - 1 RET = collection of data items
  - CollectionOfCards
    - 2 DET = 2 atts
    - 2 RET = 1-N aggregation + self-nature (collection of data items)

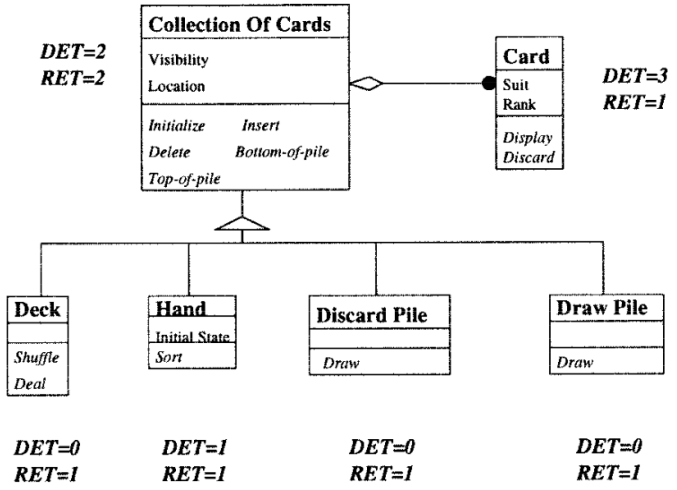


Table I. ILF and SR complexity contribution (S = Single Class, A = Aggregation, G = Generalization/Specialization, M = Mixed).

	S	A	G	M
Collection of Cards	Low	Low	–	–
Card	Low	–	Low	–
Deck	Low	Low	Low	Low
Hand	Low	Low	Low	Low
Discard Pile	Low	Low	Low	Low
Draw Pile	Low	Low	Low	Low
ILF OOFP	42	35	35	28
SR OOFP	48	48	48	48
Total OOFP	90	83	83	76

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de estimación orientados a objetos
    - Puntos Función Orientados a Objetos (OOFP)
    - Puntos Casos de Uso (UCP)
    - Puntos Clase
  - Buenas prácticas y lecciones aprendidas

- **Estimación de tamaño: Puntos Caso de Uso**
  - Técnica basada en los Puntos Función para estimar el tamaño y esfuerzo de una aplicación a partir del análisis de sus casos de uso y los actores de los mismos (G. Karner, 1993)
  - Considera además ciertos factores tecnológicos y del entorno y los computa todos en una ecuación.

Karner, Gustav. "Resource Estimation for Objectory Projects." Objective Systems SF AB, September 17, 1993.

  - ± 9% desviación en ↑95% de 200 proyectos en 5 años con una duración media de 60 meses

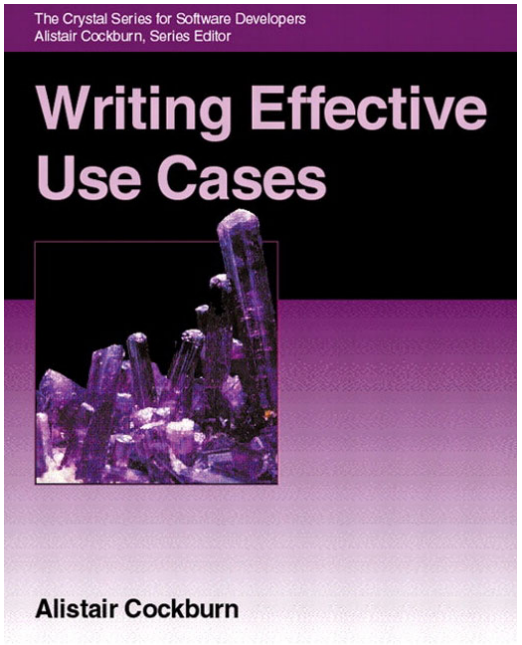
FPT

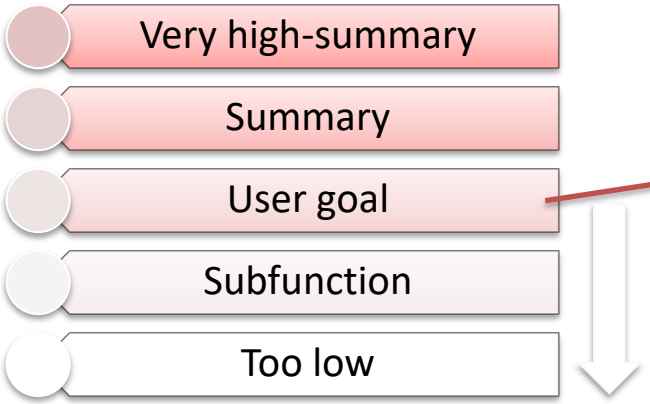
Agilis SOLUTIONS

Carrol, E.R. "Estimating Software Based on Use Case Points. OOPSLA 2005

- **Casos de Uso**
  - Un caso de uso es una secuencia de transacciones que son desarrolladas por un sistema en respuesta a un evento que inicia un **actor** sobre el propio sistema y que retorna un resultado de valor para el actor.
    - Actor
      - Toda entidad externa al sistema que guarda una relación con este y que le demanda una funcionalidad.

- **Casos de Uso**
  - Los diagramas no son suficiente, es necesaria la descripción textual
  - De 3 a 9 pasos en el escenario principal
  - Cortos, claros y fáciles de leer





Cada caso de uso es la unidad mínima de valor de negocio
- Si el usuario alcanza el objetivo, se produce una entrega de valor
- El UC se completa en una única sesión, de manera que una vez alcanzado el objetivo, el usuario puede pasar a otra actividad

- Casos de Uso

CÓDIGO DEL CASO DE USO	UC_001
NOMBRE	Nombre del caso de uso
DESCRIPCIÓN	Descripción del caso de uso
ACTOR PRINCIPAL	Es el actor que recurre a los servicios del sistema para cumplir un objetivo.
PERSONAL INVOLUCRADO E INTERESES	El caso de uso captura todo y sólo lo que satisfaga los intereses del personal involucrado. En este apartado se enumeran al personal involucrado (incluido actor principal) y sus intereses.
PRECONDICIONES	Las precondiciones establecen lo que siempre debe cumplirse antes de comenzar un escenario en el caso de uso. Normalmente una precondición implica un escenario de otro caso de uso que se ha completado con éxito.
POSTCONDICIONES	También conocidas como *garantías de éxito*. Establecen qué debe cumplirse cuando el caso de uso se completa con éxito (o bien el escenario principal o algún camino alternativo). La garantía debería satisfacer las necesidades de todo el personal involucrado.
FLUJO BÁSICO	También recibe el nombre de *escenario principal de éxito*. Describe el camino de éxito típico que satisface los intereses del personal involucrado: 1. 2. 3. 4.

- Casos de Uso

FLUJOS ALTERNATIVOS	Indican todos los otros escenarios o bifurcaciones de éxito o fracaso. Una extensión tiene dos partes: la condición y el manejo. Al final del manejo de la extensión, el escenario se une de nuevo con el escenario principal de éxito.
	2.a Condición 1. 2.
	4.a *Condición* 1. 1.a *Condición* 1. 2.
REQUISITOS ESPECIALES	En esta sección se especifican los requisitos no funcionales asociados a este caso de uso: de rendimiento, fiabilidad, etc.
LISTA DE TECNOLOGÍA Y VARIACIONES DE DATOS	A menudo puede haber variaciones técnicas en *cómo* se debe realizar algo, pero no en *qué*. Esta sección debe recoger estas variaciones técnicas (por ejemplo, una restricción tecnológica de entrada o salida de datos).
TEMAS ABIERTOS	En este apartado se reflejan todas las dudas relativas al caso de uso, que queden por resolver.

Técnicas de Estimación
Puntos Caso de Uso

- Casos de Uso

Use Case Title: Pay for a job posting
Primary actor: Recruiter
Precondition: The job information has been entered but is not viewable.
Postcondition: Job is posted; recruiter's credit card is charged.

Main Success Scenario:

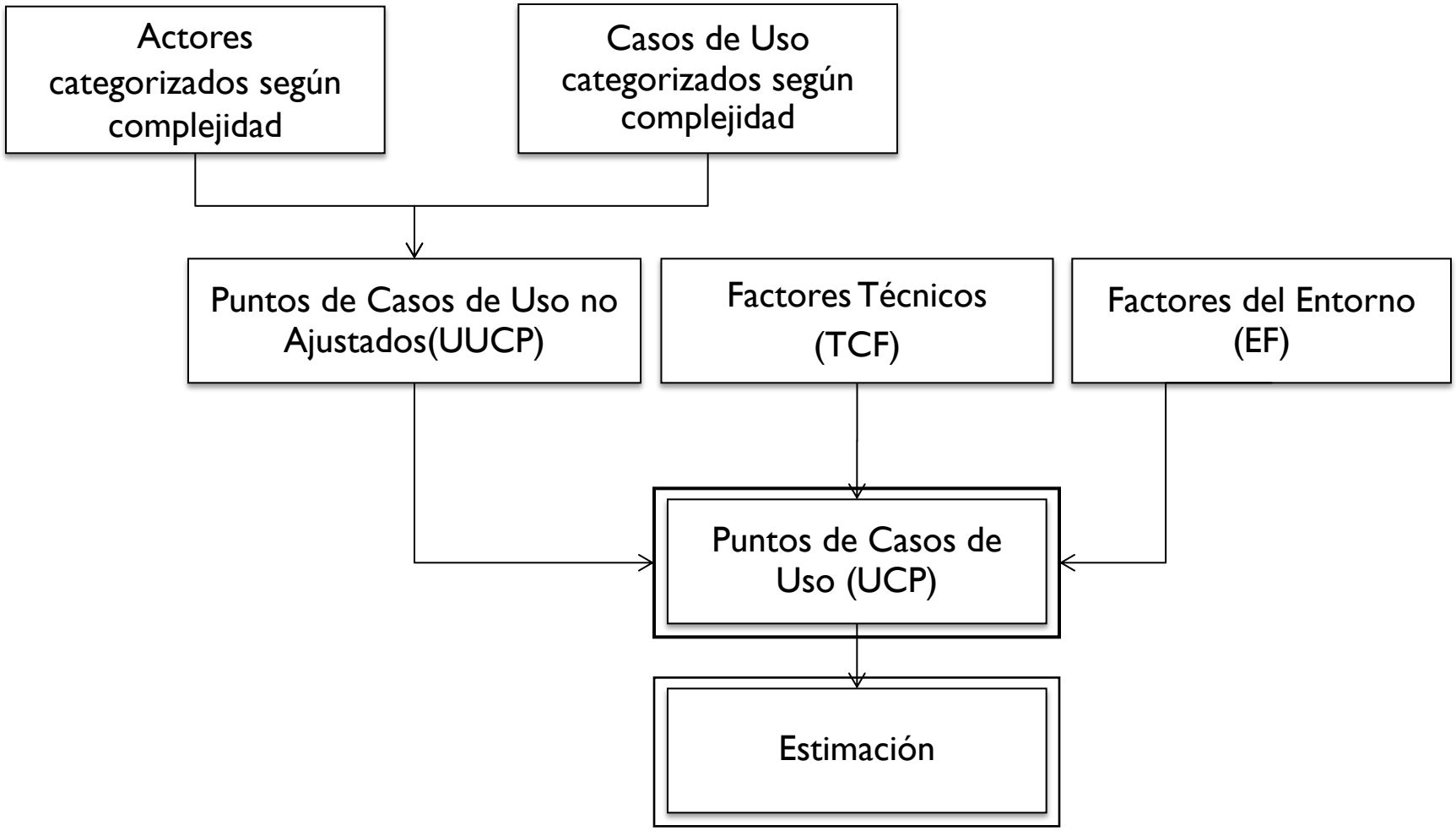
1. Recruiter submits credit card number, date, and authentication information.
2. System validates credit card.
3. System charges credit card full amount.
4. Job posting is made viewable to Job Seekers.
5. Recruiter is given a unique confirmation number.

Extensions:

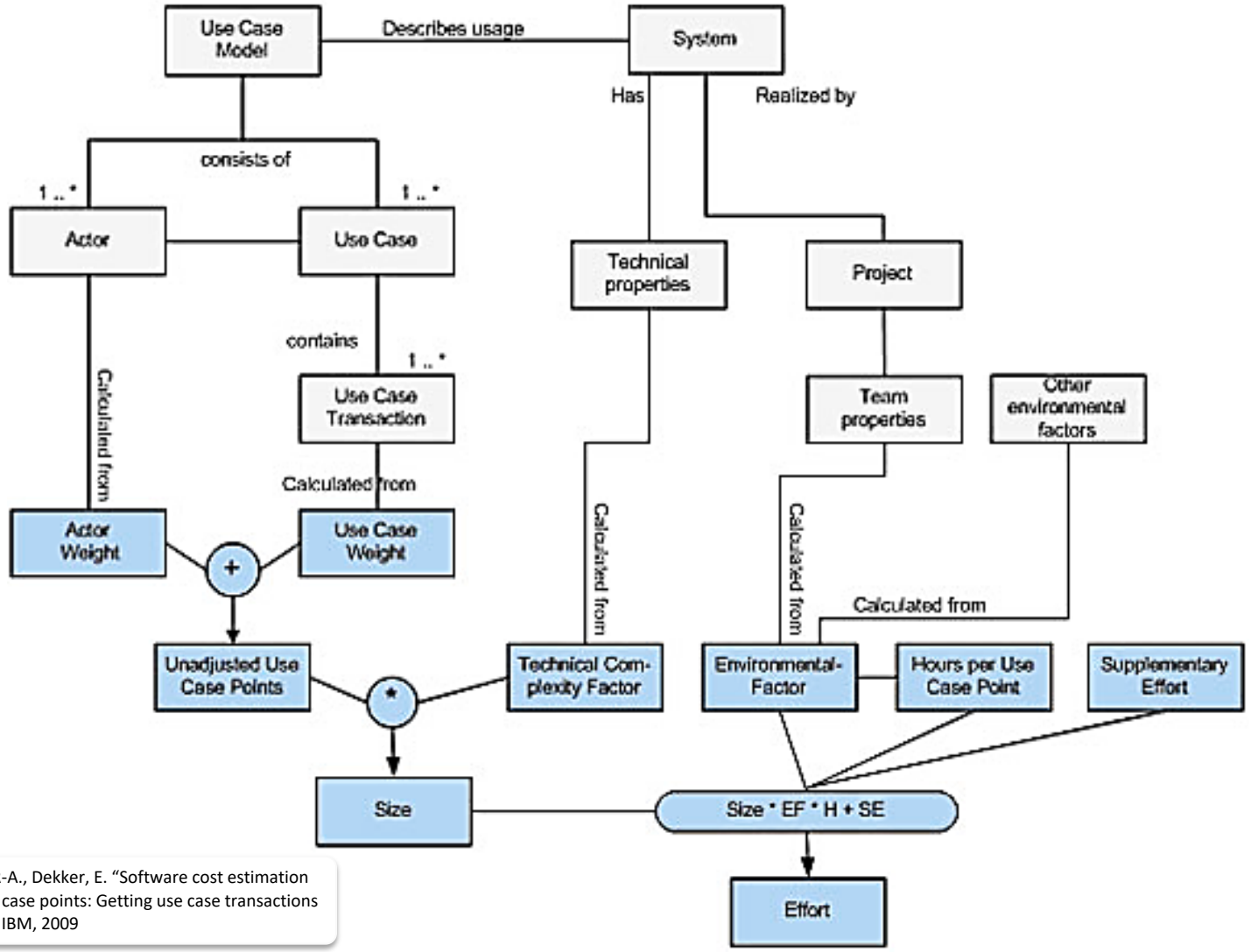
2a: The card is not of a type accepted by the system:
    2a1: The system notifies the user to use a different card.
2b: The card is expired:
    2b1: The system notifies the user to use a different card.
2c: The card number is invalid:
    2c1: The system notifies the user to re-enter the number.
3a: The card has insufficient available credit to post the ad.
    3a1: The system charges as much as it can to the current credit card.
    3a2: The user is told about the problem and asked to enter a second credit card for the remaining charge. The use case continues at Step 2.

Path = Scenario
1 -5
1, 2, 2a, 2a1, 2, 3, 4, 5
(…)

Técnicas de Estimación
Puntos Caso de Uso

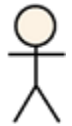


Técnicas de Estimación
Puntos Caso de Uso



Collaris, R-A., Dekker, E. "Software cost estimation using use case points: Getting use case transactions straight". IBM, 2009

- **Puntos Caso de Uso No Ajustados**



  - 1º) Cada actor* categorizado según complejidad y peso

Tipo de Actor	Descripción	Peso (Factor)
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API)	1
Medio	Otro sistema interactuando mediante un protocolo (ej.TCP/IP) o una persona interactuando a través de una interfaz en modo texto	2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica (GUI)	3

(*) → Se miden más las interacciones que los casos de uso

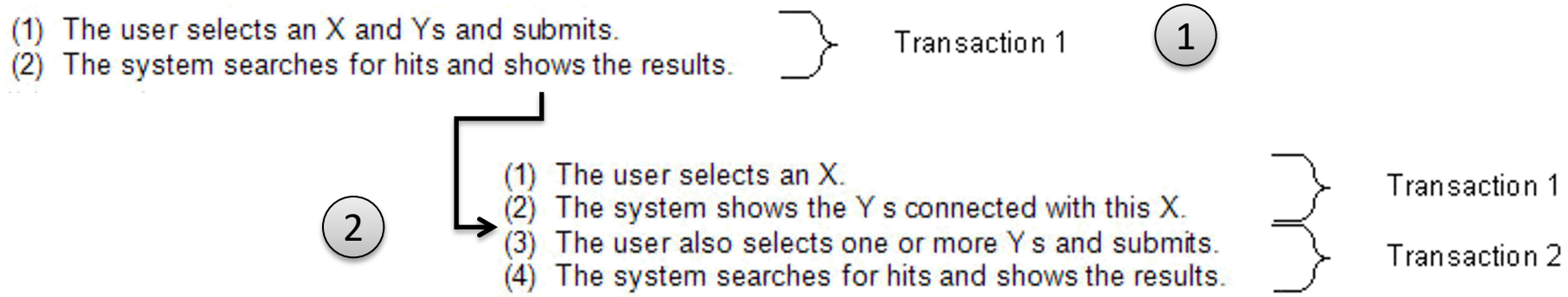
- **Puntos Caso de Uso No Ajustados**
  - 2º) La complejidad de los CU es medida en número de transacciones

Categoría de CU	Descripción	Peso (Factor)
Simple	Transacciones <= 3	5
Medio	Transacciones = [4, 7]	10
Complejo	Transacciones > 7	15

- **Transacciones**
  - Una transacción es un round-trip del usuario al sistema y vuelta al usuario
  - Una transacción no es necesariamente
    - Un paso del Caso de Uso
    - Una acción sobre la BD
  - Ejemplo: menús desplegables

El actor realiza una acción que es una entrada para el sistema
El sistema procesa esa info y devuelve un resultado al actor

¿Es esto una transacción?
El usuario pulsa la tecla 'k'
El sistema muestra una k por pantalla



- **Puntos Caso de Uso No Ajustados**
  - 3º) Los "UUCP" se calculan añadiendo los pesos por actor y caso de uso

Tipo CU	Peso	Cantidad	Resultado
Simple	5	2	10
Medio	10	2	20
Complejo	15	1	15
		Total	45

Tipo Actor	Peso	Cantidad	Resultado
Simple	1	3	3
Medio	2	3	6
Complejo	3	1	3
		Total	12

Puntos de Casos de Uso No Ajustados (UUCP)

$$UUCP = \sum_{k=1}^{n} UC_i \times w_i + \sum_{j=1}^{m} Act_j \times w_j$$

- **Factores Técnicos**
  - 4º) Los UUCP se ajustan en base a 13 factores

Factor	Descripción	Peso	Impacto *	Resultado
T1	Sistema Distribuido	2		Peso * Impacto
T2	Rendimiendo o Tiempo de respuesta	1		
T3	Eficiencia del Usuario Final	1		
T4	Procesamiento interno complejo	1		
T5	Código Reutilizable	1		
T6	Facilidad de Instalación	0.5		
T7	Facilidad de Uso	0.5		
T8	Portabilidad	2		
T9	Facilidad de cambio	1		
T10	Concurrencia	1		
T11	Características especiales de seguridad	1		
T12	Provee acceso directo a terceras partes	1		
T13	Capacidades especiales de entrenamiento	1		

Factores Técnicos (TCF)

$$0{,}6 + (0{,}01 * \sum_{r1}^{r13})$$

(*)
Irrelevante: 0 a 2
Medio: 3 a 4
Esencial: 5

- **Factores de Entorno**
  - 5º) Los UUCP se ajustan en base a 8 factores

Factor	Descripción	Peso	Impacto*	Resultado
E1	Familiaridad con el modelo de proyecto utilizado (RUP)	1.5		Peso * Impacto
E2	Personal tiempo Parcial	-1		
E3	Capacidad del analista lider	0.5		
E4	Experiencia en la aplicación	0.5		
E5	Experiencia en OO	1		
E6	Motivación	1		
E7	Dificultad del lenguaje de Programación	-1		
E8	Estabilidad de los requerimientos	2		

Factores del Entorno (EF):

$$1{,}4 + (-0{,}03 * \sum_{r1}^{r8})$$

(*)
0 a 5
1: Fuerte impacto negativo
3: Impacto Promedio
5: Fuerte impacto positivo

- Puntos Casos de Uso

Puntos de Casos de Uso (UCP) = UUCP * TCF * EF

- Productividad (Estimación Temporal)

ESTIMACIÓN = PCU * FACTOR DE PRODUCTIVIDAD

- **Factor de productividad**
  - Es la relación de horas/hombre necesaria por cada UCP
  - Si no hay datos históricos que hayan sido recabados se toman en cuenta dos posibilidades:
    - Establecer una base para los cálculos de los PCU de proyectos completados anteriormente
    - Utilizar un valor entre 15 y 30 dependiendo de la experiencia de logros pasados del equipo de desarrollo.
    - Si se trata de un nuevo equipo, usar un valor de 20 para el primer proyecto

- **Ejemplo: Amazing**
  - Software para comprar libros vía Web y mediante pago con tarjeta de crédito.
  - Dicha tarjeta de crédito debe ser validada con los sistemas del banco.
  - El sistema debe tener fuertes medidas de seguridad y debe ser de muy fácil uso (por lo demás un sistema *normal*).
  - El sistema será construido por personal externo con conocimientos técnicos medios.
  - Calcular los años/hombre de desarrollo, considerando 20 y 15 horas por punto caso de uso, jornadas de 8 horas y 20 jornadas por mes.

- Ejemplo: Amazing
  - Definición del Caso de Uso

Use Case Name: **COMPRAR LIBROS**

Short description: **el usuario podrá comprar libros y el sistema confirmará la compra.**

Basic flow of events:

1. **El usuario introduce su login y password**
2. **Selecciona el libro a comprar**
3. **El usuario selecciona las unidades del libro a comprar**
4. **El usuario introduce su número de tarjeta de crédito**
5. **El sistema valida la tarjeta de crédito con el centro de validación del Banco**
6. **El sistema muestra una confirmación**

Alternative flow of events:

3.1 No hay suficientes unidades del libro:

3.1.1 El sistema informa que no se puede realizar la compra.

5.1 La tarjeta de crédito no es valida

6.1.1 El sistema informa al usuario de que la tarjeta no es valida

6.1.2 El cliente puede cancelar la operación o repetirla.

- Ejemplo: Amazing
  - Solución
    - 1. Actores y Complejidad
      - "Usuario" - Complejidad: Alta (3)
      - "Centro de Validación de Tarjeta" - Complejidad: Media (2)
    - 2. Caso de Uso y Complejidad
      - "Comprar Libros" - Complejidad: Media (10)
    - 3. Puntos de Casos de Uso no Ajustados (UUCP)

$$UUCP = \sum_{k=1}^{n} UC_i \times w_i + \sum_{j=1}^{m} Act_j \times w_j$$

UNADJUSTED USE CASE POINTS (UUCP) = 3 + 2 + 10 = 15

- Ejemplo: Amazing
  - 4. Factores Técnicos

$$0,6 + (0,01 * \sum_{r1}^{r13})$$

Technical Factors					
Factor	Description	Weight	Influence *(0 = no influence; 3 = average; 5=strong)*	Resultant	
T1	Distributed System	2	3	6	
T2	Response adjectives	2	3	6	
T3	End-user efficiency	1	3	3	
T4	Complex processing	1	3	3	
T5	Reusable code	1	3	3	
T6	Easy to install	0,5	3	1,5	
T7	Easy to use	0,5	5	2,5	
T8	Portable	2	3	6	
T9	Easy to change	1	3	3	
T10	Concurrent	1	3	3	
T11	Security features	1	5	5	
T12	Access for third parties	1	3	3	
T13	Special training required	1	3	3	
				48	TFactor

Complexity Factor (TCF)	1,08

- Ejemplo: Amazing
  - 5. Factores Entorno

$$1{,}4 + (-0{,}03 * \sum_{r1}^{r8})$$

Enviroment Factors				
Factor	Description	Weight	Influence *(0 = no influence; 3 = average; 5 strong)*	Resultant
F1	Familiar with RUP	1,5	3	4,5
F2	Application experience	0,5	3	1,5
F3	Object-oriented experience	1	3	3
F4	Lead analyst capability Motivation	0,5	3	1,5
F5	Motivation	1	3	3
F6	Stable requirements	2	3	6
F7	Part-time workers	-1	5	-5
F8	Difficult programming languaje	-2	3	-6
				8,5 EFactor

Environmental Factor	1,145

- Ejemplo: Amazing

UCP = UUCP * TCF * EF = 15 * 1,08 * 1,145 = 18,549

Podemos comparar proyectos,
"éste es más complicado que …"

- **Estimación de esfuerzo**
  - Aplicar la tasa media de productividad
  - Los factores de productividad dependen de la organización … lo importante es que todo el mundo utilice el mismo factor y método
  - Dos aproximaciones
    - Karner: 20 h/UCP
    - Schneider and Winters
      - 20 h/UCP proyectos simples » $\left(\sum_{i=1}^{i=6}(E_i|E_i<3)+\sum_{j=7}^{j=8}(E_j|E_j>3)\right)>3$
      - 28 h/UCP proyectos complejos » $\left(\sum_{i=1}^{i=6}(E_i|E_i<3)+\sum_{j=7}^{j=8}(E_j|E_j>3)\right)>5$

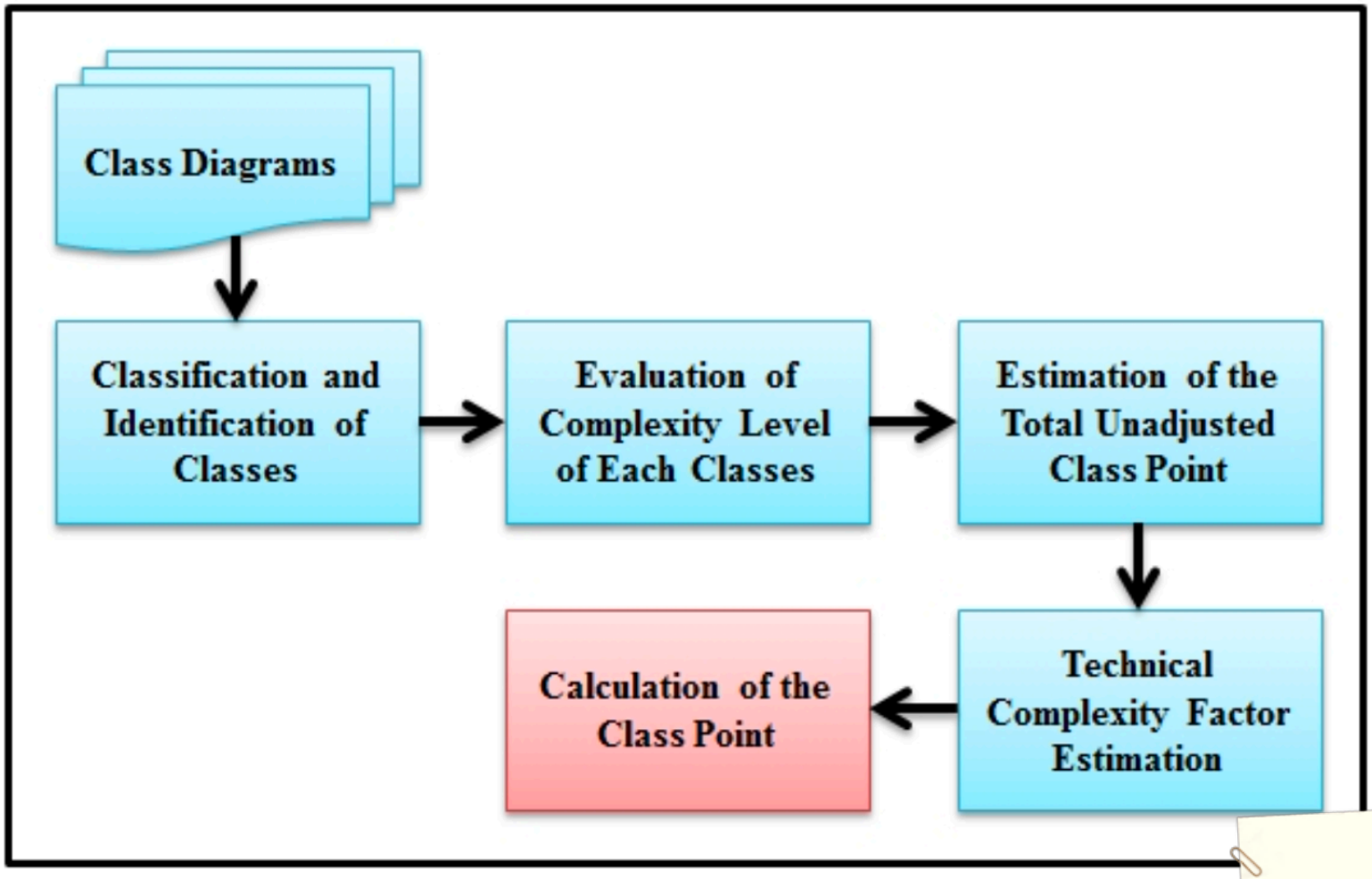
Schneider, G.,Winters, J.P., Applied Use Cases, Second Edition, A Practical Guide, Addison-Wesley, Reading, MA, 2001

Si el valor es > 5 reconsiderar el proyecto

- **Estimación de Tamaño: Puntos Caso de Uso**
  - Ventajas
    - POO
    - Puede llegar a automatizarse
  - Desventajas
    - No aplica en fases demasiado tempranas
    - No existe un estándar para documentar UCs
    - Igual que los Puntos Función, no se adapta bien a proyectos de mantenimiento
    - Algunos factores técnicos no tienen realmente un impacto en todo el proyecto
      - Facilidad de instalación …(Historias de Usuario ad-hoc?)

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de estimación orientados a objetos
    - Puntos Función Orientados a Objetos (OOFP)
    - Puntos Casos de Uso (UCP)
    - Puntos Clase
  - Buenas prácticas y lecciones aprendidas

Puntos Clase



[Costaglioa et al., 2005]
[Yucalar et al., 2013]

- **Identificar y clasificar clases**
  - Problem Domain Type (PDT)
    - Representan entidades del mundo real
  - Human Interaction Type (HDT)
    - Soportan la visualización de información o interacción con el usuario
  - Data Management Type (DMT)
    - Almacenamiento y consulta de datos
  - Task Management Type (TMT)
    - Tareas de control y definición

- **Asignar complejidades**
  - CP_1: estimación preliminar de tamaño
    - Number of External Methods (NEM)
      - Tamaño de la interfaz de la clase: métodos definidos localmente
    - Number of Service Requests (NSR)
      - Volumen de interconexiones de la clase: peticiones a otras clases
  - CP_2: al disponer de más información se refina la estimación preliminar
    - Se añade el Number of Attributes (NOA)

CP_1 de una clase

	0 – 4 NEM	5 – 8 NEM	≥ 9 NEM
0 – 1 NSR	Low	Low	Average
2 – 3 NSR	Low	Average	High
≥ 4 NSR	Average	High	High

0 – 2 NSR	0 – 5 NOA	6 – 9 NOA	≥ 10 NOA
0 – 4 NEM	Low	Low	Average
5 – 8 NEM	Low	Average	High
≥ 9 NEM	Average	High	High

(a)

3 – 4 NSR	0 – 4 NOA	5 – 8 NOA	≥ 9 NOA
0 – 3 NEM	Low	Low	Average
4 – 7 NEM	Low	Average	High
≥ 8 NEM	Average	High	High

(b)

≥ 5 NSR	0 – 3 NOA	4 – 7 NOA	≥ 8 NOA
0 – 2 NEM	Low	Low	Average
3 – 6 NEM	Low	Average	High
≥ 7 NEM	Average	High	High

CP_2 de una clase

- **Cálculo de Puntos Clase No Ajustados (UCP)**
  - Una vez identificados el tipo y complejidad de cada clase, sabemos el valor con el que deben contribuir a la ecuación

Component Type	Description	Complexity			
		Low	*Average*	*High*	*Total*
PDT	Problem Domain Type	… x 3 = …	… x 6 = …	… x 10 = …	…
HIT	Human Interaction Type	… x 4 = …	… x 7 = …	… x 12 = …	…
DMT	Data Management Type	… x 5 = …	… x 8 = …	… x 13 = …	…
TMT	Task Management Type	… x 4 = …	… x 6 = …	… x 9 = …	…
				Total Unadjusted Class Point (TUCP):	

$$UCP = \sum_{i=1}^{4} \sum_{j=1}^{3} x_{ij} * w_{ij}$$

- # Factor de Complejidad Técnico
  - Influencia (de 0 a 5) de 18 características del sistema

- Not present or no influence = 0
- Insignificant influence = 1
- Moderate influence = 2
- Average influence = 3
- Significant influence = 4
- Strong influence = 5

ID	System Characteristic	DI	ID	System Characteristic	DI
C1	Data communication	…	*C10*	Reusability	…
C2	Distributed functions	…	*C11*	Installation ease	…
C3	Performance	…	*C12*	Operational ease	…
C4	Heavily used configuration	…	*C13*	Multiple sites	…
C5	Transaction rate	…	*C14*	Facilitation of change	…
C6	Online data entry	…	*C15*	User adaptivity	…
C7	End-user efficiency	…	*C16*	Rapid prototyping	…
C8	Online update	…	*C17*	Multiuser interactivity	…
C9	Complex processing	…	*C18*	Multiple interfaces	…
				Total Degree of Influence (TDI):	

$$F = 0.55 + (0.01 * TDI)$$

$$CP = UCP * F$$

- Ejemplo

Characteristic	Project OFDS
Size	3 months elapsed time, 4630 Lines of Code
Programming Environment	Java, MySQL, Eclipse, ObjectAid UML Class Diagram Plugin
Project Members	3 developers with 0 to 13 years' experience
Application Domain	Commercial

System Component	Number of Classes
PDT	6
HIT	5
DMT	3
TMT	3

Component Type	Description	Complexity			
		Low	*Average*	*High*	*Total*
PDT	Problem Domain Type	6 x 3 = 18	0 x 6 = 0	0 x 10 = 0	18
HIT	Human Interaction Type	1 x 4 = 4	2 x 7 = 14	2 x 12 = 24	42
DMT	Data Management Type	2 x 5 = 10	1 x 8 = 8	0 x 13 = 0	18
TMT	Task Management Type	2 x 4 = 8	1 x 6 = 6	0 x 9 = 0	14
				Total Unadjusted Class Point (TUCP):	**92**

$TDI = 47$

$F = 0.55 + (0.01 * TDI)$ $\quad F = 1{,}02$

$CP = UCP * F$ $\quad CP = 92 * 1{,}02 = 93{,}84$

$Size = CP \times Prog.Lang.LOC\ Parameter$ $\quad Size = 93{,}84 * 46 \cong 4317\ LOC$

- Estimación de Proyectos Software
  - Introducción
  - Métodos de Estimación
    - Métodos Heurísticos
    - Métodos Paramétricos
    - Métodos Algorítmicos o Empíricos
  - Métodos de estimación orientados a objetos
    - Puntos Función Orientados a Objetos (OOFP)
    - Puntos Casos de Uso (UCP)
    - Puntos Clase
  - Buenas prácticas y lecciones aprendidas

- **Rangos de Estimaciones**
  - Permite realizar estimaciones razonables a través de una media ponderada de los tres factores del rango

[Eoptimista, Eprobable, Epesimista]

LDC = (LDCopt + 4 LDCprob + LDCpes)/6

1.- Estimación de las LDC.

Funciones	LDC_{opt}	LDC_{pro}	LDC_{pes}	LDC
Consultas catálogo	3.500	4.300	6.000	4.450
Préstamos y devoluciones	2.000	2.400	3.100	2.450
Mantenimiento catálogo	1.100	1.700	2.400	1.717
Servicios Web	2.000	2.800	3.500	2.783
	Estimación LDC total			11.400

- **Rangos de Estimación (II)**
  - Comparación de Modelos empíricos de estimación
    - Lo adecuado es utilizar dos o más técnicas y posteriormente, comparar los resultados
    - Se toman todas las estimaciones y se realiza un rango de estimación, con un valor mínimo, uno medio más realista y uno máximo

Estimación aceptable $Valor_{min} - Valor_{máx} \leq 20\ \%$

	Estim. A (LDC)	Estim. B (PF)	Estim. C (Tareas)	Media	Rango	Difer. (%)
Coste (€)	45.600 €	48.000 €	52.160 €	48.600 €	[45.600-48.580-52.160]	14,4 %
Esfuerzo (pm)	14,25 pm	14,00 pm	16,30 pm	14,93 pm	[14,00 –14,93-16,30]	16,4 %

- **Enfoque Recomendado**
  - En general, no es bueno ceñirse a un sólo método de estimación
  - Primeras estimaciones: apoyarse en el juicio de expertos con técnica Delphi.
    - Que los expertos constituyan un grupo específico más o menos permanente.
  - Con **especificaciones detalladas** ya se puede estimar su tamaño aplicando técnicas de descomposición y modelos algorítmicos.
  - Conviene aplicar modelos o ecuaciones locales, es decir, no tomar sin más modelos desarrollados en otros entornos. Es preferible adaptarlos o crear ecuaciones propias:
    - Desarrollar modelos y procedimientos de estimación apropiados a las características de nuestra organización.
      - P.e: parámetros fácilmente medibles en nuestra organización …
    - Recoger datos de los proyectos desarrollados y crear procedimientos de estimación que puedan cambiarse a medida que se analizan los datos recogidos.
  - Conviene mejorar y refinar continuamente los métodos y las ecuaciones, informando a los expertos de los resultados obtenidos y corrigiendo las fórmulas.

- **Enfoque Recomendado**
  - Si creamos un modelo o ecuación propio
    - Especificar modelo de proceso e identificar actividades a estimar
    - Formular teoría: elegir y relacionar medidas básicas con el coste
      - Podemos partir de una ecuación conocida
    - Analizar los datos y mejorar y refinar el modelo
  - Uso de herramientas CASE
    - Implementación de modelos conocidos (o propios)
    - Extracción de métricas base

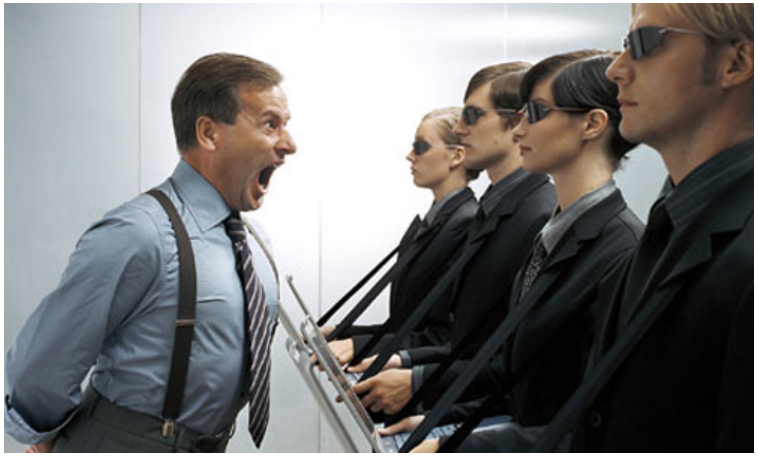
Cada organización debe acumular datos de sus proyectos para conocer mejor su productividad y los factores que influyen en ella (que rara vez coinciden con los de otra empresa) y poder estimar mejor sus proyectos

Buenas prácticas

- Estimación de proyecto **≠ Estimación de producto**
  - Formación, Instalación, Pasos a producción, Vacaciones, Reuniones, Enfermedades, etc.
- Evitar estimaciones improvisadas, dedicar tiempo a estimar
- Usar datos de proyectos anteriores y comparar con datos históricos.
- Revisar estimaciones periódicamente y/o utilizar diferentes técnicas de estimación a la vez que el proyecto avanza
- Utilizar rangos de forma gradual
- Estimar por módulos
- Ser pesimista

- **Plus-or-minus qualifiers** "6 months, +3 months, -2 months"
- **Ranges** "5-9 months"
- **Risk quantification** "6 months...+1 month for late subcontractor, +0.5 month for staff sickness, etc..."
- **Coarse dates and time periods** "3rd quarter 97"
- **Confidence factors**
  - April 1 5%
  - May 1 50%
- **Cases**
  - Best case April 1
  - Planned case May 15
  - Worst case July 15

- **Errores comunes**
  - Estimar demasiado tiempo antes de saber el qué
  - Crear una estimación para un nuevo proyecto comparándolo con uno anterior…
    - En el que se trabajó fines de semana hasta las 4 a.m., etc.
    - Utilizando la estimación para el anterior, en lugar de los valores reales
  - Asumir que la mejor estimación es del que tiene mejores cuerdas vocales (p.e. departamento de ventas).
  - Crear estimaciones olvidando formación, reuniones, fiestas, vacaciones, tiempos del cliente.
  - No considerar los riesgos

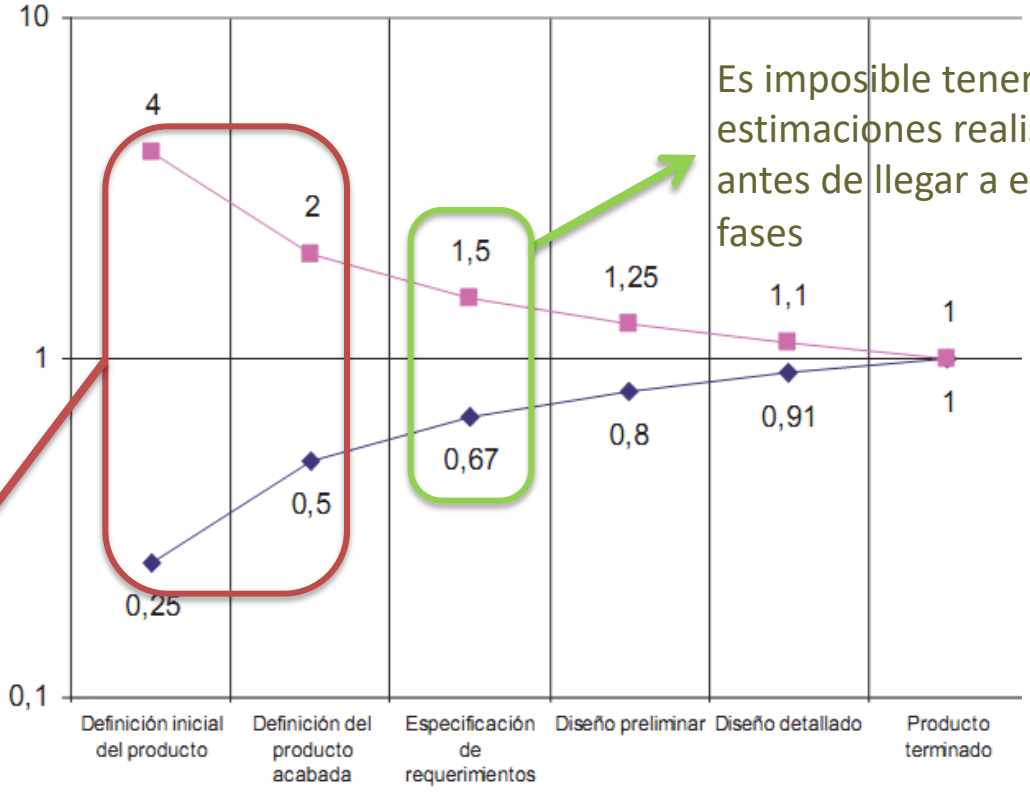


- **Errores Comunes (II)**
  - Confundir objetivos y estimaciones
  - Decir "Si" cuando deberíamos decir "No"
    - Implica confianza en tus estimaciones
  - Hacer compromisos demasiado pronto en el cono de incertidumbre
  - Asumir que la infra-estimación tiene un impacto neutral
  - Perseguir la Ciencia de la Estimación en lugar del Arte de la Estimación
  - Estimar en la "zona imposible"
  - Sobre-estimar el ahorro introducido por nuevos métodos o herramientas
  - Utilizar una única técnica de estimación

- Hacer compromisos demasiado pronto en el cono de incertidumbre

Cuanta más información tengamos, mejor será la estimación

A medida que avanza el proyecto, mejora la precisión de la estimación

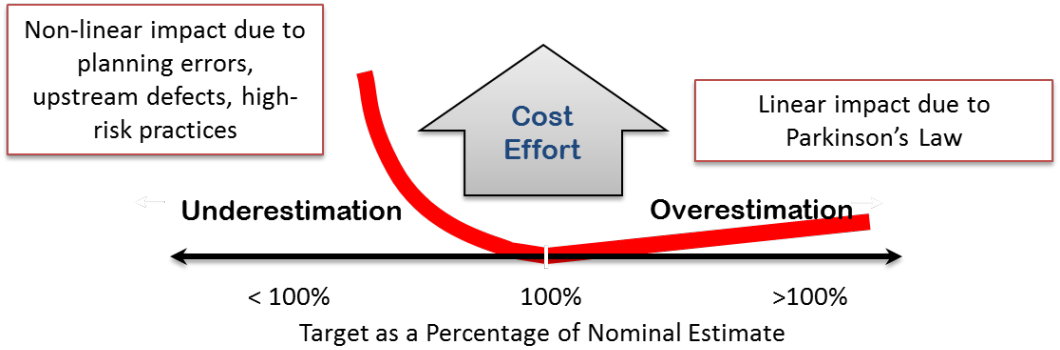


Es imposible tener estimaciones realistas antes de llegar a estas fases

Una estimación hecha en la fase de definición inicial o de definición final del producto tiene un rango de fallo de entre 2x – 4x

El cono representa el **mejor caso**
Es imposible ser más exacto (tal ver más afortunado)

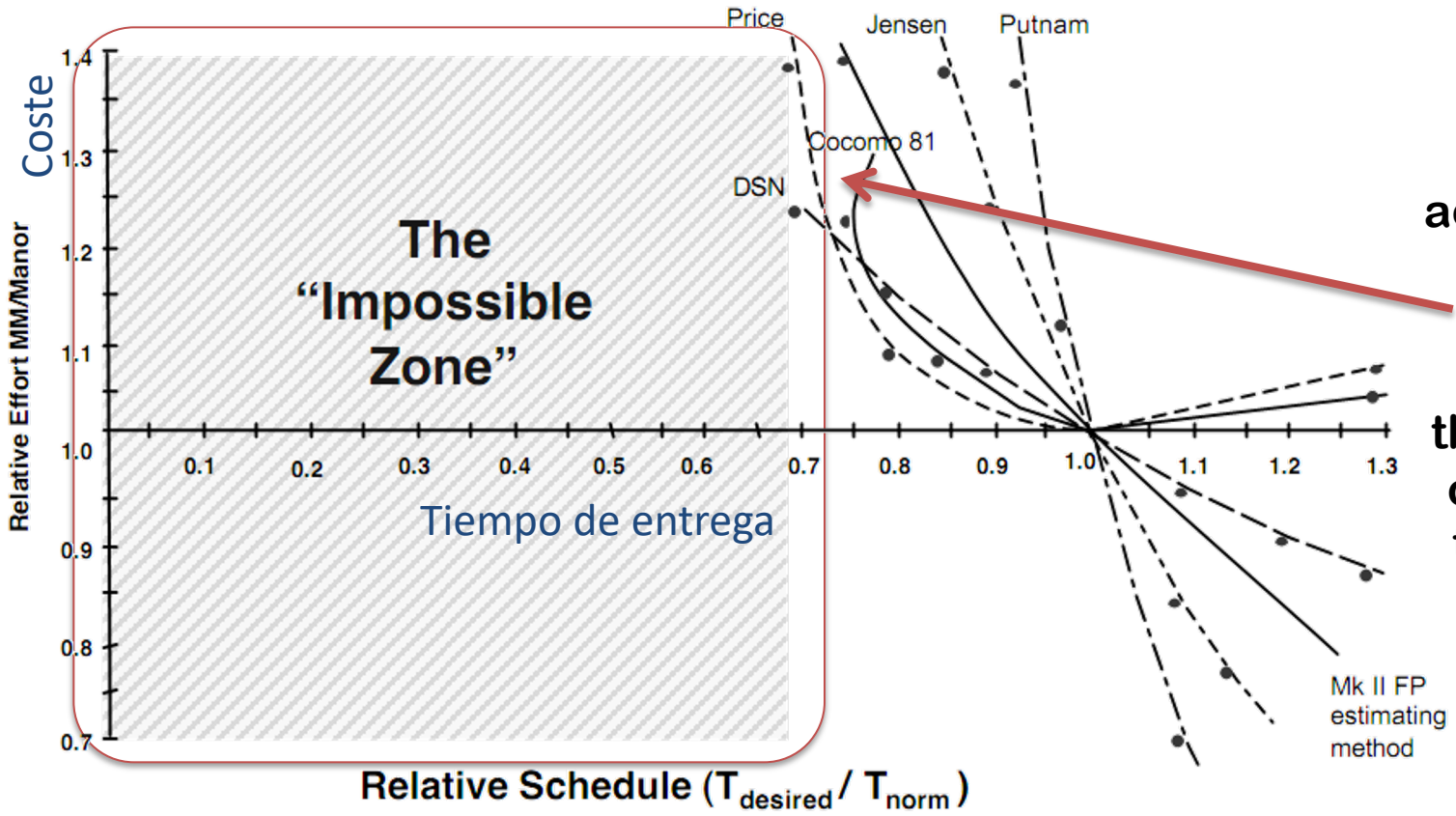
- **Asumir que estimar a la baja no tiene impacto**
  - Sobre-estimar:
    - Ley de parkinson: el trabajo se "expande"
    - Sindrome de Goldratt: procrastinamos hasta el final y acabamos por incumplir el dead-line



[McConnell, 2005]

  - Infra-estimar: los costes se disparan
    - Reduce la efectividad del plan de proyecto (basado en falacias)
    - Reduce estadísticamente las posibilidades de éxito (on-time)
    - Menos tiempo para actividades iniciales (requisitos, diseño, etc.)
      - Modificaciones a posteriori, que siempre resultan más caras y complejas
    - Actividades para reconducir el proyecto
      - Reuniones re-planificación; Versiones "intermedias" para demos, etc; Reparar apaños que se introdujeron por la presión del calendario (*quick and dirty workarounds*) …

- Estimar en la "zona imposible"
  - Hay un límite en el factor de comprensión alrededor del 25%



The cost of achieving the efficient schedule is much less than the cost of achieving the shortest possible schedule.

[McConnell, 2005]

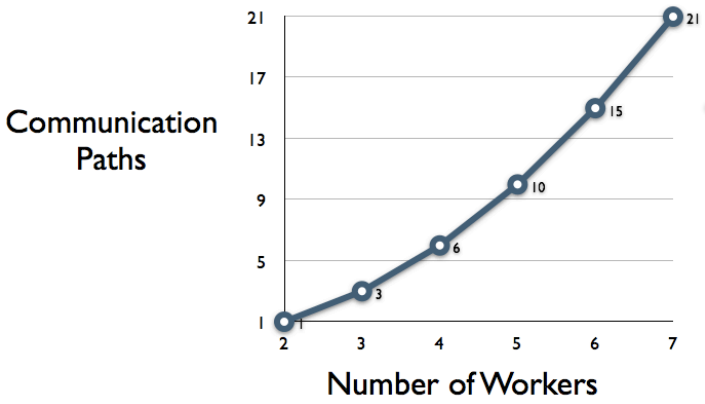
Errores comunes

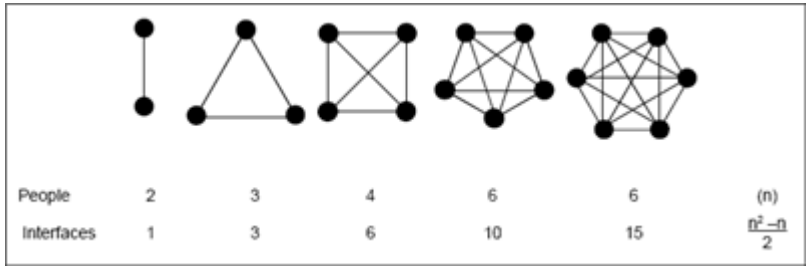
Brook's Law: Adding manpower to a late software project makes it later

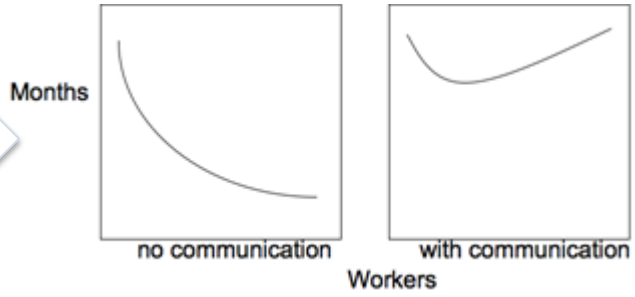
- Medir el progreso de un proyecto en función del tiempo que lleva desarrollándose es un error
- Personas y tiempo no son intercambiables

Ejemplo

- Si un programador codifica un programa de 1 KLOC en una semana …
- Pueden 5 programadores codificarlo en un día?
- Y 40 hacerlo en una hora?







Dilbert's on estimation

© Scott Adams, Inc./Dist. by UFS, Inc.

Referencias

- [Antoniol et al., 1999] Antoniol, G., Lokan, C., Caldiera, G., & Fiutem, R. (1999). A function point-like measure for object-oriented software. *Empirical Software Engineering*, *4*(3), 263-287.
- [Boehm, 1981] Boehm, B., Software Engineering Economics, Prentice-Hall, 1981.
- [Boehm et al., 2000] Barry W. Boehm, Clark, Horowitz, Brown, Reifer, Chulani, Ray Madachy, and Bert Steece. 2000. Software Cost Estimation with Cocomo II (1st ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.
- [Costagliola et al., 2005] Costagliola, G., Ferrucci, F., Tortora, G., & Vitiello, G. (2005). Class point: an approach for the size estimation of object-oriented systems. *IEEE Transactions on Software Engineering*, *31*(1), 52-74.
- [McConnell, 1996] Rapid Development - Taming Wild Software Schedules. Microsoft Press, 1996.
- [McConnell, 2006] Software estimation - Demystifying the black art. Microsoft Press, 2006.
- [Piattini, 2008] Piattini, M., García, F., Garzás, J. y Genero, M. "Medición y Estimación del Software". Ra-Ma, 2008.
- [Pressman, 2010] Ingeniería del Software. Un Enfoque Práctico (7ª Edición) Pressman, R. S. Editorial : McGraw-Hill (2010)
- [Putnam, 1997] Putnam, L., y W. Myers, «How Solved is the Cost Estimation Problem», IEEE Software, Noviembre 1997.
- [Sommerville, 2005] Sommerville, I. Ingeniería del Software. McGraw-Hill (7ª Edición)
- [Yucalar et al., 2013] Yucalar, F., Borandag, E., & Erdogan, S. Z. (2013, January). SIZE ESTIMATION USING CLASS POINT METHOD FOR OBJECT-ORIENTED SYSTEMS. In International Symposium on Computing in Science & Engineering. Proceedings (p. 123). GEDIZ University, Engineering and Architecture Faculty.