

## Resumen Ingeniería de Software 2: clase 5 - Métricas

### *Recordando: elementos clave de la gestión de proyectos*

- **Métricas.**
- **Estimaciones.**
- Calendario temporal.
- Organización del personal.
- Análisis de riesgos.

### **Métricas**

- Es la clave tecnológica para el desarrollo y el mantenimiento de software.
  - Objetivos fundamentales de las métricas:
    - Entender que ocurre en el desarrollo/mantenimiento.
    - Controlar desarrollo/mantenimiento.
    - Mejorar nuestro producto de software.
    - Evaluar la calidad del mismo.

### **Definiciones de métrica**

- Medida: indicación cuantitativa de un algo.
  - Ej: cuantas personas necesito para hacer mi proyecto, que cantidad de líneas de código tiene mi producto terminado.
- Medición: es el acto de medir, de determinar la medida, es el cálculo para llegar a la medida.
- Métrica: lo que yo utilizo para lograr la medida.
  - Para realizar la medición aplico una métrica y el resultado es una medida.
- Indicador: es una combinación de métricas, nos define si estamos bien encaminados en un proceso o si debemos hacer algún ajuste.

### **Métricas**

- Las métricas pueden ser utilizadas para que profesionales e investigadores puedan tomar las mejores decisiones.
- Las métricas se pueden dividir para asegurar la calidad en procesos/proyectos de software / productos.

### **Métricas del proyecto**

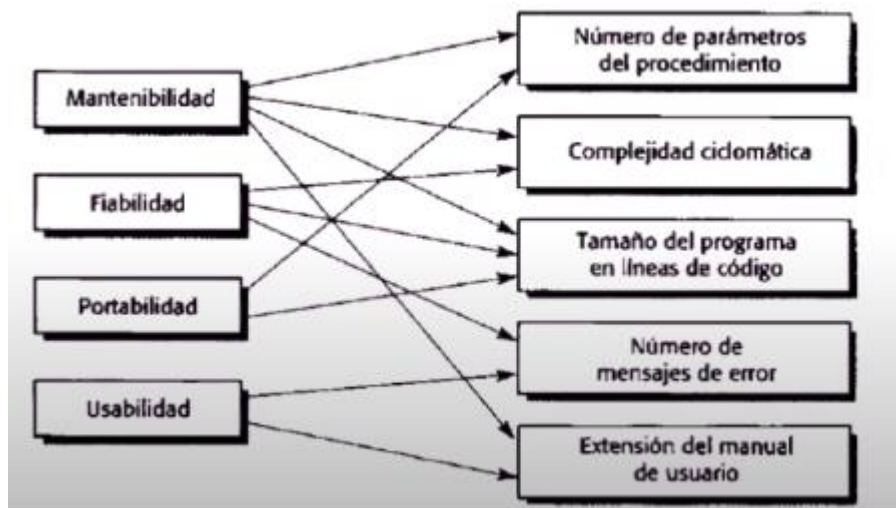
- Usos:
  - Ajustes en el calendario y evitar demoras.
  - Valorar el estado de un proyecto en marcha.
  - Rastrear riesgos.
  - Descubrir áreas de problemas.

## Resumen Ingeniería de Software 2: clase 5 - Métricas

- Ajustar flujo de trabajo/tareas.
- Evaluar habilidad del equipo.

### **Métricas del proceso**

- Tiene dos propósitos estratégicos:
  - Recopilar información a través de todos los proyectos y por un espacio de tiempo.
  - La intención es proporcionar un conjunto de indicadores para mejorar el proceso.
- Usos:
  - Incorporar al grupo un sentido común y una sensibilidad organizacional.
  - Retroalimentaciones.
  - No se usan las métricas para valorar a los individuos sobre otros.
  - Para establecer metas y métricas claras.
  - No excluir métricas, sino ampliarlas.
- Atributos de calidad externos vs Atributos internos.



### **Métricas del producto**

- Se dividen en dos:

## Resumen Ingeniería de Software 2: clase 5 - Métricas



### dinámicas

- Hechas en un programa en ejecución.
- Ayudan a valorar la eficiencia y fiabilidad de un programa.



### estáticas

- Hechas de representaciones del sistema
- Ayudan a valorar la complejidad, comprensibilidad y mantenibilidad.

○

### Métricas estáticas del producto

Fan-in/Fan-out	Fan-in es una medida del número de funciones o métodos que llaman a otra función o método (por ejemplo, X). Fan-out es el número de funciones que son llamadas por una función X. Un valor alto de fan significa que X está fuertemente acoplada al resto del diseño y que los cambios en X tendrán muchos efectos importantes. Un valor alto de fan-out sugiere que la complejidad de X podría ser alta debido a la complejidad de la lógica de control necesaria para coordinar los componentes llamados.
Longitud del código	Ésta es una medida del tamaño del programa. Generalmente, cuanto más grande sea el tamaño del código de un componente, más complejo y susceptible de errores será el componente. La longitud del código ha mostrado ser la métrica más fiable para predecir errores en los componentes.
Complejidad ciclomática	Ésta es una medida de la complejidad del control de un programa. Esta complejidad del control está relacionada con la comprensión del programa.
Longitud de los identificadores	Es una medida de la longitud promedio de los diferentes identificadores en un programa. Cuanto más grande sea la longitud de los identificadores, más probable será que tengan significado; por lo tanto, el programa será más comprensible.
Profundidad del anidamiento de las condicionales	Ésta es una medida de la profundidad de anidamiento de las instrucciones condicionales «if» en un programa. Muchas condiciones anidadas son difíciles de comprender y son potencialmente susceptibles de errores.
Índice de Fog	Ésta es una medida de la longitud promedio de las palabras y las frases en los documentos. Cuanto más grande sea el Índice de Fog, el documento será más difícil de comprender.

### Métricas del producto – LDC – Líneas de Código

- Es la métrica más común para el tamaño de un producto, a su vez es una métrica muy discutida.
  - Es una métrica postmortem, ya que se usa cuando el producto está terminado (ya que sabemos cuántas líneas de código tiene el proyecto).
- Resultados que nos otorga:
  - Tiempo usado.
  - Cuántas personas se involucraron.

## Resumen Ingeniería de Software 2: clase 5 - Métricas

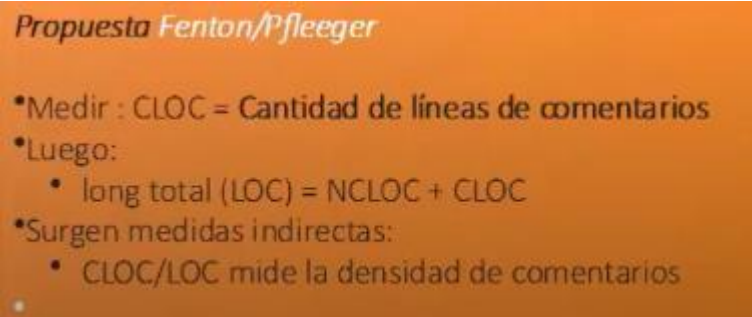
- Si una organización de software mantiene registros sencillos de proyectos anteriores similares, podemos calcular y aplicar las métricas del LDC.

### **Utilidad de las métricas postmortem**

- Se conforma una línea base para futuras métricas. La línea base la da un proyecto que es el primero de todos (como un paciente cero), es importante que dicho proyecto nos haya salido bien.
- Las métricas postmortem ayudan al mantenimiento conociendo la complejidad lógica, tamaño, flujo de información, identificación de módulos críticos.
- Ayuda en los procesos de reingeniería.

### **Métricas orientadas al tamaño**

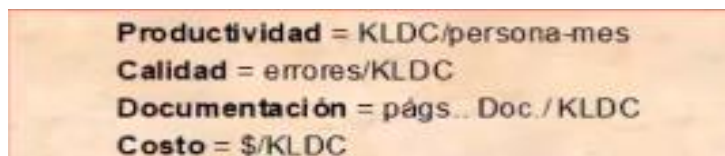
- Podemos obtener de ellas:
  - Productividad: relación entre **KLDC (miles de líneas de código)** / Persona més.
  - Calidad: relación entre errores / KLDC.
  - Costo: relación entre \$ /KLDC
- Como manejo: líneas en blanco, comentarios, etc.



**Propuesta Fenton/Pfleeger**

- \*Medir : CLOC = Cantidad de líneas de comentarios
- \*Luego:
  - long total (LOC) = NCLOC + CLOC
- \*Surgen medidas indirectas:
  - CLOC/LOC mide la densidad de comentarios

- 
- Ejemplo de cálculo usando LDC:



**Productividad** = KLDC/persona-mes  
**Calidad** = errores/KLDC  
**Documentación** = págs. Doc./KLDC  
**Costo** = \$/KLDC

○

## Resumen Ingeniería de Software 2: clase 5 - Métricas

Proyecto	LDC	U\$S	Errores	Personas-mes	Errores/KLDC	U\$S/KLDC	KLDC/Personas-mes
P1	25.500	15000	567	15	22,23 %	588,23	1,7
P2	19.100	7200	210	10	10,99 %	376,96	1,91
P3	10.700	6000	100	20	9,34 %	560,74	0,53
P4	100.000	18000	2200	30	22 %	180	3,33

○

### Métrica de Punto función:

- Mide la cantidad de funcionalidad de un sistema descrito en una especificación.

### PF- Punto función (Albrecht 1978)

$$PF = TOTAL * [0.65 + 0.01 * SUM(F_i)] \quad i=1 \text{ a } 14 \quad 0 \leq F_i \leq 5$$

	simple	medio	complejo	
# Entradas	*	[ 3   4   6 ]	=	.....
# Salidas	*	[ 4   5   7 ]	=	.....
# Consultas	*	[ 3   4   6 ]	=	.....
# Almacenamientos internos	*	[ 7   10   15 ]	=	.....
# Interfaces externas	*	[ 5   7   10 ]	=	.....
				TOTAL

➤



## Resumen Ingeniería de Software 2: clase 5 - Métricas

### PF- Punto función (Albrecht 1978)

Factor de Ponderación, es subjetivo y esta dado por la organización/equipo

$$PF = TOTAL * [0.65 + 0.01 * SUM(F_i)] \quad i=1 \text{ a } 14 \quad 0 \leq F_i \leq 5$$

	simple	medio	complejo	
# Entradas	* [ 3   4   6 ]	=	.....	
# Salidas	* [ 4   5   7 ]	=	.....	
# Consultas	* [ 3   4   6 ]	=	.....	
# Almacenamientos internos	* [ 7   10   15 ]	=	.....	
# Interfaces externas	* [ 5   7   10 ]	=	.....	
				TOTAL

Son valores de ajuste de la complejidad según las preguntas de la siguiente pantalla

1. ¿Requiere el sistema copias de seguridad y de recuperación fiables?
2. ¿Se requiere comunicación de datos?
3. Existen funciones de procesamiento distribuido?
4. ¿Es crítico el rendimiento?
5. ¿Se ejecuta el sistema en un entorno operativo existente y fuertemente utilizado?
6. ¿Requiere el sistema entrada de datos interactiva?
7. ¿Requiere la entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas u operaciones?
8. ¿Se actualizan los archivos maestros de forma interactiva?
9. ¿Son complejas las entradas, las salidas, los archivos o las peticiones?
10. ¿Es complejo el procesamiento interno?
11. ¿Se ha diseñado el código para ser reutilizable?
12. ¿Están incluidas en el diseño la conversión y la instalación?
13. ¿Se ha diseñado el sistema para soportar múltiples instalaciones en diferentes organizaciones?
14. ¿Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizada por el usuario?

0	1	2	3	4	5
No-influencia	Incidental	Moderado	Medio	Significativo	Esencial

Sum(F<sub>i</sub>) = sumatoria del puntaje de cada pregunta.

- 
- Métricas derivadas:

$$\begin{aligned} \text{Productividad} &= PF / \text{Persona\_mes} \\ \text{Calidad} &= \text{Errores} / PF \\ \text{Costo} &= \$ / PF \end{aligned}$$

- Es una medida más subjetiva independiente del lenguaje (a las líneas de código), de estimación más fácil, además es una métrica temprana.

### Desarrollo de una métrica – GQM (OPM)

- Victor Basili desarrolló un método llamado GQM (Goal, Question, Metric) (o en castellano: OPM Objetivo, Pregunta, Métrica).

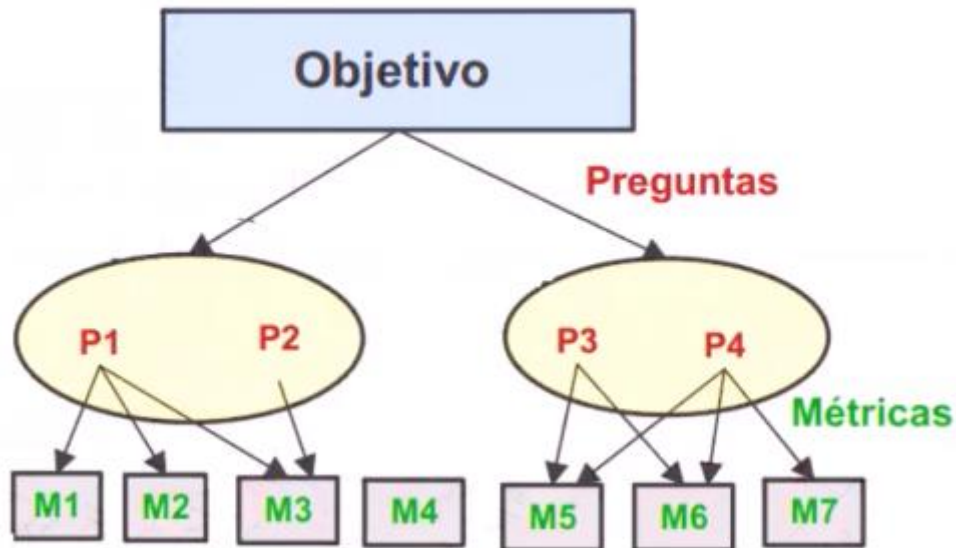
## Resumen Ingeniería de Software 2: clase 5 - Métricas

- Dicho método está orientado a lograr una métrica que “mida” cierto objetivo. El mismo nos permite mejorar la calidad de nuestro proyecto.

### GQM (OPM)

#### ➤ Estructura:

- Nivel Conceptual (Goal / Objetivo):
  - Se define un objetivo para el proyecto.
- Nivel Operativo (Question / Pregunta):
  - Se refina un conjunto de preguntas a partir del objetivo, con el propósito de verificar su cumplimiento.
- Nivel Cuantitativo (Metric / Métrica):
  - Se asocia un conjunto de métricas para cada pregunta, de modo de responder a cada una de un modo cuantitativo.



- Es útil para decidir qué medir.
- Debe estar orientado a metas.
- Es flexible.