MÉTRICAS DE SOFTWARE

Presentado por:
Jhon Alexander Guamanga
Cyntia Mileidy Herrera
Andrés Felipe Orejuela

¿Qué son las métricas de software?

El concepto de métrica es el término que describe muchos y muy variados casos de medición. Siendo una métrica una medida estadística, estas medidas son aplicables a todo el ciclo de vida del desarrollo, desde la iniciación, cuando debemos estimar los costos, al seguimiento y control de la fiabilidad de los productos finales, y a la forma en que los productos cambian a través del tiempo debido a la aplicación de mejoras. Un ingeniero del Software recopila medidas y desarrolla métricas para obtener indicadores.

¿Qué son las métricas de software? (Cont.)

En la mayoría de los desafíos técnicos, las métricas nos ayudan a entender tanto el proceso técnico que se utiliza para desarrollar un producto, como el propio producto. *El producto se mide para intentar aumentar su calidad*.

Las métricas no son absolutas ni son comprobaciones científicas sólidas.

Proporcionan una manera sistemática de evaluar la calidad a partir de un conjunto de reglas definidas con claridad.

¿Qué son las métricas de software? (Cont.)

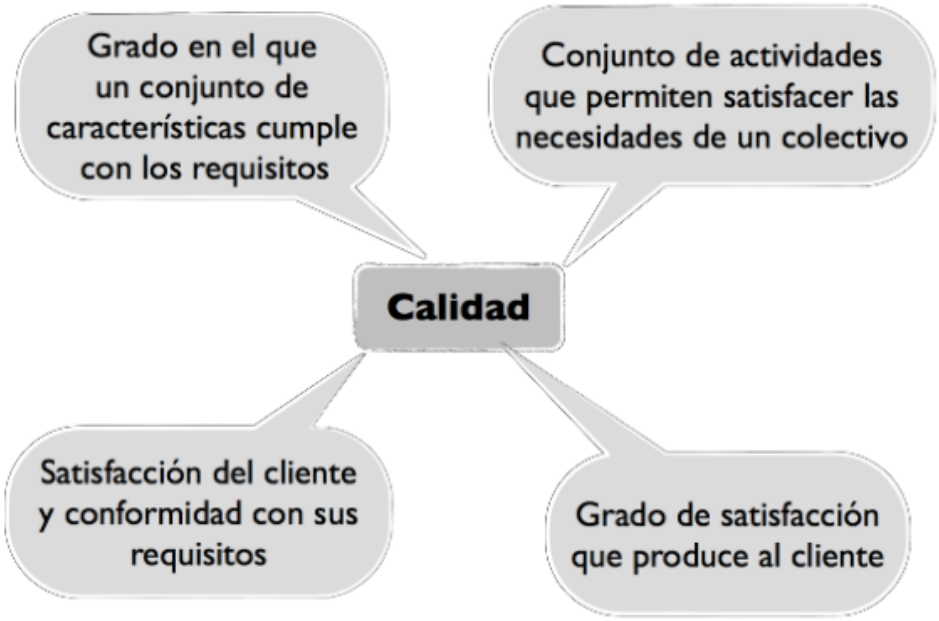
En general, la medición persigue tres objetivos fundamentales: ayudarnos a entender qué ocurre durante el desarrollo y el mantenimiento, permitirnos controlar qué es lo que ocurre en nuestros proyectos y poder mejorar nuestros procesos y nuestros productos (Fenton y Pfleeger, 1997).

RAZONES PARA MEDIR UN PRODUCTO

- Para indicar la calidad del producto.
- Para evaluar la productividad de la gente que desarrolla el producto.
- Par evaluar los beneficios en términos de productividad y de calidad, derivados del uso de nuevos métodos y herramientas de la ingeniería de software.
- Para establecer una línea de base para la estimación.
- Para ayudar a justificar el uso de nuevas herramientas o de formación adicional.

RAZONES PARA MEDIR UN PRODUCTO (Cont.)

En otras palabras las métricas de software van íntimamente relacionadas con la calidad:



ANTECEDENTES

- La métricas de software son un intento de cuantificar todos los aspectos de los productos de software incluidos en el código del programa, la especificación funcional, diseño de sistemas y diseño detallado.
- Las métricas de software realmente comenzó a principios de los años ochenta con el trabajo realizado por dos académicos de la Universidad de Iowa, Kafura oennis y Sally Henry. Ellos trataron de investigar el diseño del sistema métrico que podría ser utilizado para predecir factores tales como la facilidad de mantenimiento.

UTILIDADES

- Estimar casos de prueba
- Ayudar a entender rangos de productividad amplios
- Ayudar a entender el crecimiento de Proyectos
- Ayudar a calcular el costo real del software
- Estimar el costo de proyectos, la programación y el esfuerzo
- Ayudar a entender los costos de mantenimiento
- Ayudar con las negociaciones de contrato

CATEGORÍAS

Aunque se ha propuesto una gran variedad de taxonomías métricas, las siguientes atienden las áreas más importantes de las métricas.

- Métricas para el modelo de análisis
- Métricas para el modelo de diseño
- Métricas para el código fuente
- Métricas para pruebas

MÉTRICAS PARA EL MODELO DE ANÁLISIS

En esta fase las métricas técnicas proporcionan una visión interna a la calidad del modelo de análisis. Estas métricas examinan el modelo de análisis con la intención de predecir el tamaño del sistema resultante. Es probable que el tamaño y la complejidad del diseño estén directamente relacionadas.

MÉTRICAS PARA EL MODELO DE ANÁLISIS

Métricas basadas en la función

La métrica de punto de función (PF) es para medir la funcionalidad que entrega un sistema. Se usa para:

1. estimar el costo o el esfuerzo requerido para diseñar, codificar y probar el software
2. predecir el número de errores que se encontrarán durante la prueba
3. pronosticar el número de componentes, de líneas de código proyectadas, o ambas, en el sistema implementado.

MÉTRICAS DE PUNTO DE FUNCIÓN

Los valores del dominio de la información se definen de la siguiente manera:

- # Entradas Externas (EE) : Usuario, aplicación modifica ALI
- # Salida Externa (SE): Informes, mensajes de error, pantallas
- # Consultas Externas (CE): Recuperación de datos de uno o más archivos lógicos internos y de archivos externos de la interfaz

MÉTRICAS DE PUNTO DE FUNCIÓN (Cont.)

- # Archivos Lógicos Internos (ALI): Archivos lógicos internos que se mantienen a través de entradas externas.
- # Archivos Interfaz Externos (ALE): Datos lógicamente relacionados que se utilizan para procesos de referencia

Componente	Complejidad Total		
	Bajo	Medio	Alto
Número de Entradas Externas (EE) (EI)	63	13	2
Número de Salidas Externas (EO) (SE)	10	0	0
Número de Consultas Externas (EQ) (CE)	19	0	0
Número de Archivos Lógicos Internos (ILF) (ALI)	8	7	0
Número de Archivos de Interfaz Externos (EIF) (AEI)	0	0	0

MÉTRICAS DE PUNTO DE FUNCIÓN (Cont.)

Cálculo de los puntos de función sin ajustar:

Una vez señalado identificadores, su complejidad y modificadores se asignan pesos que ya están dados por el método.

Puntos de Función sin ajustar.				
Indicador	simple	mediano	complejo	SUMA
ALI	_ * 7	_ * 10	_ * 15	___
AIE	_ * 5	_ * 7	_ * 10	___
EE	_ * 3	_ * 4	_ * 6	___
SE	_ * 4	_ * 5	_ * 7	___
CE	_ * 3	_ * 4	_ * 6	___
			T =	___

Componente	Complejidad Total			
	Bajo	Medio	Alto	Total
Número de Entradas Externas (EE) (EI)	189	52	12	253
Número de Salidas Externas (EO) (SE)	40	0	0	40
Número de Consultas Externas (EQ) (CE)	57	0	0	57
Número de Archivos Lógicos Internos (ILF) (ALI)	56	70	0	126
Número de Archivos de Interfaz Externos (EIF) (AEI)	0	0	0	0
Conteo total				476

Puntos de función sin ajustar: 476.

MÉTRICAS DE PUNTO DE FUNCIÓN (Cont.)

Cálculo Punto de Función

$$PF_{estimado} = conteo_{total} * (0.65 + (0.01 * TDI))$$

El Grado Total de Influencia (TDI) es la sumatoria de los pesos dados a los factores de ajuste. Se califican entre [0-5]: 0=sin influencia, 1=accidental, 2=moderado, 3=medio, 4=significativo, 5=esencial.

MÉTRICAS DE PUNTO DE FUNCIÓN (Cont.)

Factor de Ajuste		Descripción	Peso
1	Comunicación de Datos	¿Cuantas facilidades de comunicación hay disponibles para ayudar con el intercambio de información con la aplicación o el sistema?	4
2	Procesamiento distribuido de los datos	"Distribuida" significa que los componentes (o los datos) de la aplicación están distribuidos en dos o más procesadores diferentes (esto también incrementa el factor anterior). ¿Cómo se manejan los datos y las funciones de procesamiento distribuido?	3
3	Rendimiento	¿Existen requerimientos de velocidad o tiempo de respuesta?	4
4	Configuraciones fuertemente utilizadas	¿Qué tan intensivamente se utiliza la plataforma de hardware donde se ejecutará la aplicación o el sistema?	2
5	Tasas de Transacción	¿Con qué frecuencia se ejecutan las transacciones? diarias, semanales, mensuales...	4
6	Entrada de datos On-line	¿Qué porcentaje de la información se ingresa on-line?	0
7	Diseño para la eficiencia de usuario final	¿se designa la aplicación para maximizar la eficiencia del usuario final?	4
8	Actualizaciones on-line	¿Cuantos archivos lógicos internos se actualizan por una transacción on-line?	0
9	Procesamiento complejo	¿Hay procesamientos lógicos o matemáticos intensos en la aplicación?	1
10	Reusabilidad	La aplicación se desarrolla para suplir una o muchas de las necesidades de los usuarios	5
11	Facilidad de instalación	¿Es muy difícil la instalación y la conversión al nuevo sistema?	4
12	Facilidad de operación	¿Cómo de efectivos y automatizados son los procedimientos de arranque, parada, backup y restore del sistema?	2
13	Puestos Múltiples	¿La aplicación fue concebida para su instalación en múltiples sitios y organizaciones?	5
14	Facilidad de cambio	La aplicación fue concebida para facilitar los cambios sobre la misma	2
GRADO TOTAL DE INFLUENCIA (TDI)			40

$$PF_{estimado} = 476 * (0.65 + (0.01 * 40)) = 476 * 1.05 = 499.8 \approx 500$$

MÉTRICAS DE PUNTO DE FUNCIÓN (Cont.)

Por ejemplo para calcular los costos en tiempo y dinero:

Esfuerzo horas/persona	$= PF/(\frac{1}{8}\ persona/hora)$	4000 horas/persona
Duración proyecto en horas:	$\frac{\text{Esfuerzo horas/persona}}{\#\ personas\ (en\ este\ caso\ 4)}$	1000 horas por miembro
Duración proyecto en meses:	$\frac{\text{Duración proyecto en horas}}{100\ horas/mes}$	10 meses

MÉTRICAS DE PUNTO DE FUNCIÓN (Cont.)

Teniendo en cuenta que el salario mensual en promedio por persona está en: $1’700’000, entonces para un grupo de 4 personas, 20 días laborables y 5 horas productivas, el costo total estimado del proyecto es de $68,000,000.00 (1’700’000 * 10 * 4) de pesos colombianos; y a esto es le suman los costos de transporte, viajes, hardware, etc. lo que da el costo total del proyecto.

MÉTRICAS PARA EL MODELO DE DISEÑO

Proporcionan al diseñador una mejor visión interna. Ayudan a que el diseño evolucione a un nivel superior de calidad.

Éstas se concentran en las características de la estructura del programa dándole énfasis a la estructura arquitectónica y en la eficiencia de los módulos. Estas métricas son de caja negra, en el sentido de que no se requiere ningún conocimiento del trabajo interno de ningún modo en particular del sistema.

CLASIFICACIÓN MÉTRICAS PARA EL MODELO DE DISEÑO

- Métricas arquitectónicas
- Métricas al nivel de componente
- Métricas de diseño de la interfaz
- Métricas especializadas en diseño orientado a objetos

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

Card y Glass definen tres medidas de la complejidad del diseño del software basados en la complejidad de los módulos:

La complejidad estructural S(i), de un módulo i se define de la siguientes manera.

$$S(i) = f^2_{out}(i) \qquad (4.12)$$

donde fout (i) es la dependencia hacia fuera del modulo i.

de módulos que son invocados directamente por el módulo i

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

La complejidad de datos D(i), proporciona una indicación de la complejidad en la interfaz interna de un módulo i, y se define como :

$$D(i) = v(i) / [f_{out}(i) + 1] \qquad (4.13)$$

donde v(i) es el número de variables de entrada y salida que se pasan a o se reciben del módulo i.

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

Finalmente. **la complejidad del sistema C(i)**, se define como la suma de las complejidades estructural y de datos, y se define como.

$$C(i)=S(i)+D(i) \qquad (4.14)$$

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

A medida que crecen los valores de complejidad, la complejidad arquitectónica o global del sistema también aumenta. Esto lleva a una mayor probabilidad de que aumente el esfuerzo necesario para la integración y las pruebas.

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

En 1991,Fenton sugiere varias métricas de morfología simples que permiten comparar diferentes arquitecturas de programa mediante un conjunto de dimensiones directas.

- Tamaño = n + a, n es el # de nodos, a es el # de arcos
- Profundidad = el camino más largo desde el nodo raíz a un nodo hoja.

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

- Anchura = máximo número de nodos de cualquier nivel de la arquitectura.
- Relación arco-a-nodo (acoplamiento de la arquitectura) = mide la densidad de conectividad de la arquitectura y puede proporcionar una sencilla indicación del acoplamiento de esta. $r = a / n$

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

Ejemplo:

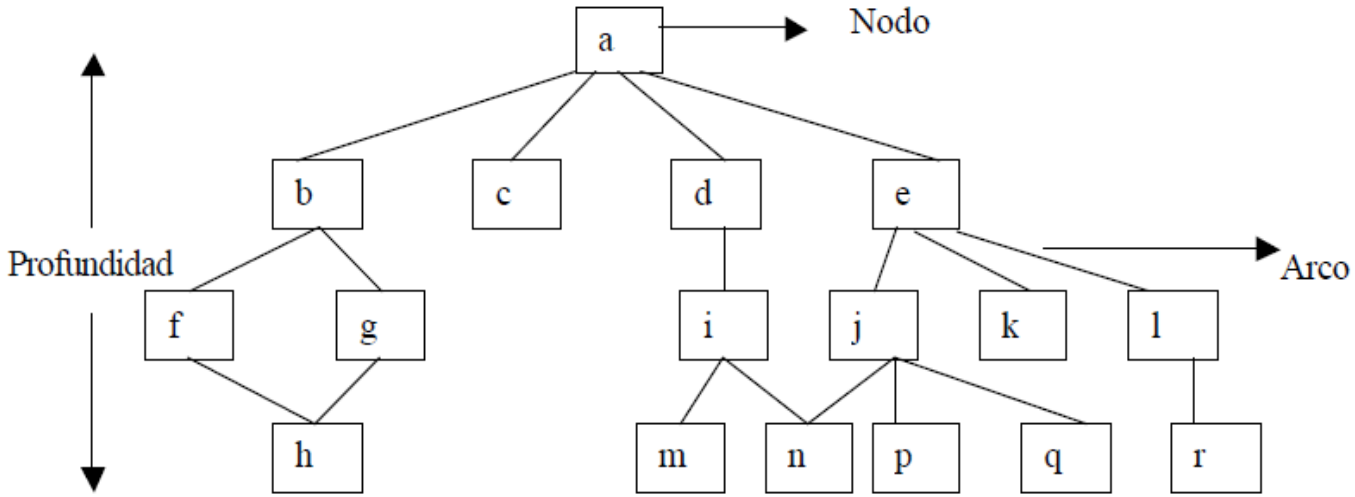


Figura 4.4 Arquitectura de Software [Fenton '91]

MÉTRICAS PARA EL MODELO DE DISEÑO

Métricas arquitectónicas

Para la arquitectura mostrada en la Figura 4.4, [Fenton ’91]

tamaño = 17 + 18 = 35

profundidad =4.

anchura = 6.

r = 18/17 = 1,06. medida sencilla del acoplamiento de la arquitectura.

MÉTRICAS PARA EL CÓDIGO FUENTE

Estás métricas asignadas como cuantitativas por Halstead, se derivan después de que se ha generado el código o se estima una vez que el diseño esté completo.

Un programa esta compuesto de “tokens”, las instrucciones del lenguaje, los identificadores, constantes, operadores delimitadores de comentario y signos especiales del mismo. De esta forma se obtiene una medida más realista de la cantidad de información contenida en el código fuente.

MÉTRICAS PARA EL CÓDIGO FUENTE (Cont.)

Las medidas son:

- n1: el número de operadores diferentes que aparecen en el programa.
- n2: el número de operandos diferentes que aparecen en el programa.
- N1: el número total de ocurrencias de operadores
- N2: el número total de ocurrencias de operandos

MÉTRICAS PARA EL CÓDIGO FUENTE (Cont.)

Aclaración:

- Los operadores son las palabras reservadas del lenguaje, tales como IF-THEN, READ, FOR,...; los operadores aritmeticos +, -, *,..... los de asignacion y los operadores logicos AND, EQUAL TO,....
- Los operandos son las variables, literales y las constantes del programa.

MÉTRICAS PARA EL CÓDIGO FUENTE (Cont.)

Longitud (N):

- Se calcula como, $N = N1 + N2$
- Es una simple medida del tamaño de un programa.
- Cuanto más grande sea el tamaño de N, mayor será la dificultad para comprender el programa y mayor el esfuerzo para mantenerlo.

Volumen (V):

- Da un peso extra al número de operadores y operandos únicos. Por ejemplo, si dos programas tienen la misma longitud N pero uno tiene mayor número de operadores y operandos únicos, que naturalmente lo hacen más difícil de entender y mantener, este tendrá un mayor volumen.
- Se calcula como $V = N \times \log_2(n)$ donde $n = n1 + n2$

MÉTRICAS PARA EL CÓDIGO FUENTE (Cont.)

Dificultad (D):

- Para definir la dificultad **D** del programa, se usa la fórmula siguiente: **D** = (n1 * N2) / (n2 *2).

Esfuerzo (E):

- El esfuerzo es otra medida estudiada por Halstead que ofrece una medida del trabajo requerido para desarrollar un programa.
- Desde el punto de vista del mantenimiento, el esfuerzo se puede interpretar como una medida del trabajo requerido para comprender un software ya desarrollado.
- La fórmula es la siguiente: **E** = D * V ó V / L
- Volumen mínimo (L): **L** = 2/n1 * n2/N2

MÉTRICAS PARA EL CÓDIGO FUENTE (Cont.)

Por ejemplo, consideremos el siguiente trozo de programa:

```
if  (N < 2)
{
    A = B * N;
    System.out.println("El resultado es : " + A);
}
```

MÉTRICAS PARA EL CÓDIGO FUENTE (Cont.)

A partir de aquí se deduce:

- Operadores Únicos **n1** = 6

(if, {}, system.out.println, =, *, <)

- Ocurrencias de Operadores **N1** = 6

(if, {}, system.out.println, =, *, <)

- Operandos Únicos **n2** = 4

(N, A, B, 2)

- Ocurrencias de Operandos **N2** = 6

(N, 2, A, B, N, A)

MÉTRICAS PARA EL CÓDIGO FUENTE (Cont.)

- Longitud (N) = 6 + 6 = 12
- Volumen (V) = 27.509 * log2(6+4) = 91.382
- Dificultad (D) = (6 * 6)/(4 * 2) = 4.5
- Esfuerzo (E) = 91.382 * 4.5 = 411.219

MÉTRICAS PARA PRUEBAS

Aunque se ha escrito mucho sobre métricas del software para pruebas, la mayoría de las métricas propuestas se concentran en el proceso de pruebas, no en las características técnicas de las pruebas mismas.

Los responsables de la prueba, se guían por las métricas de análisis, diseño y código.

- Puntos de Función
- Bang
- Halstead
- Complejidad ciclomática

MÉTRICAS PARA PRUEBAS (Cont.)

Por ejemplo:

Las métricas basadas en la función pueden emplearse para predecir el esfuerzo global de las pruebas Se pueden juntar y correlacionar varias características a nivel de proyecto (p ej.: esfuerzo y tiempo las pruebas, errores descubiertos, número de casos de prueba producidos) de proyectos anteriores con el número de Pf producidos por un equipo del proyecto. El equipo puede después predecir los valores 'esperados' de estas características del proyecto actual.

MÉTRICAS PARA PRUEBAS (Cont.)

Las métricas de diseño de alto nivel proporcionan información sobre facilidad o dificultad asociada con la prueba de integración y la necesidad de software especializado de prueba (p. ej.: matrices y controladores) La complejidad ciclomática (una métrica de diseño de componentes) se encuentra en el núcleo de las pruebas de caminos básicos, un método de diseño de casos de prueba. Además, la complejidad ciclomática puede usarse para elegir módulos como candidatos para pruebas más profundas. Los módulos con gran complejidad ciclomática tienen más probabilidad de tendencia a errores que los módulos con menor complejidad ciclomática

MÁS MÉTRICAS

Métricas Básicas:

Este grupo de métricas computan información básica sobre el código

- Lines of Code Per Method
- Number of Constructors Per Type
- Number of Fields Per Type
- Number of Methods Per Type
- Number of Parameters
- Lines of Code

MÁS MÉTRICAS

Métricas de Herencia:

Este grupo contiene métricas basadas en la estructura de herencia del código.

- Average Depth of Inheritance Hierarchy
- Average Number of Subtypes
- Depth of Inheritance Tree (DIT)
- Number of Children (NOC)

MÁS MÉTRICAS

Métricas de Complejidad:

Este grupo de métricas miden la complejidad del código:

- Cyclomatic Complexity
- Weigthed Methods per Class (WMC)
- Block Depth

MÁS MÉTRICAS

Métricas de Dependencia y Comunicación: Este grupo contiene métricas que miden la responsabilidad, independencia y estabilidad de un trozo de código.

- Abstractness
- Afferent Couplings
- Distance
- Efferent Couplings
- Instability
- Coupling between object classes (CBO)
- Response for a class (RFC)
- Lack of cohesion in methods (LCOM)

Bibliografía

[1]http://www.uv.mx/personal/asumano/files/2012/08/MetricasTecnicas.pdf

[2]http://techi322.files.wordpress.com/2008/08/metricas-del-producto-para-el-sw.pdf

[3]http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/gonzalez_d_h/capitulo4.pdf

[4]http://sistinfii.files.wordpress.com/2011/08/siii11-06-mc3a9tricas-21.pdf

GRACIAS POR SU ATENCION