Universidad Nacional de Tucumán

Universidad Nacional de Tucumán
Facultad de Ciencias Exactas y Tecnología

Ingeniería de Software II

Mediciones del Software - métricas para sistemas OO

Mg. Héctor A. Valdecantos

Sistemas orientados a objetos

- Razones para las métricas especializadas en OO
  - **Localización**: cómo se distribuye la información en un programa. El "objeto" como unidad básica del SW.
  - **Encapsulamiento**: sobre el empaquetamiento de una colección de elementos, agrupación semántica de los elementos que construyen el software.
  - **Ocultamiento de información**: cómo se oculta detalles operativos para determinadas partes del programa para mostrar sólo lo necesario.
  - **Herencia**: mecanismo de generalización-especialización de clases existente en OO.
  - **Abstracción**: sobre cómo se enfoca en lo esencial de un concepto ignorando detalles de nivel inferior.

[Berard, "Metrics for object-oriented software engineering.", (1998)]

Sistemas orientados a objetos

	Tradicional	OO
Localización	funciones, procedimientos	la unidad básica son los objetos/clases
Encapsulación	bajo nivel: registros y arreglos, mid-level: funciones, procedimientos, subrutinas	mecanismos más amplios de encapsulación: clase, paquetes y módulos. Objetos encapsulan estado y métodos, interfaz; opcionalmente otros objetos, constantes; y más importante “conceptos”
Ocultamiento de Información	inclusión de archivos (header files), Function and Procedure Scope, variables locales o estáticas,	mecanismo especializados: private, protected, public, como ejemplo en C++
Herencia	~N/A	herencia simple o múltiple: número de subclases clases, de clases bases, nivel de profundidad de herencia
Abstracción	de función, de datos, de procesos	de objeto. Entidades de alto nivel como cajas negras. Clases y clases paramétricas. Número de instancias por clases (C), nro de clases parametrizadas (CP), la proporción entre CP/C

[resumen de: Berard, Edward V. "Metrics for object-oriented software engineering." The Object Agency, Inc (1998)]

Métricas Clásicas Orientadas a Clases

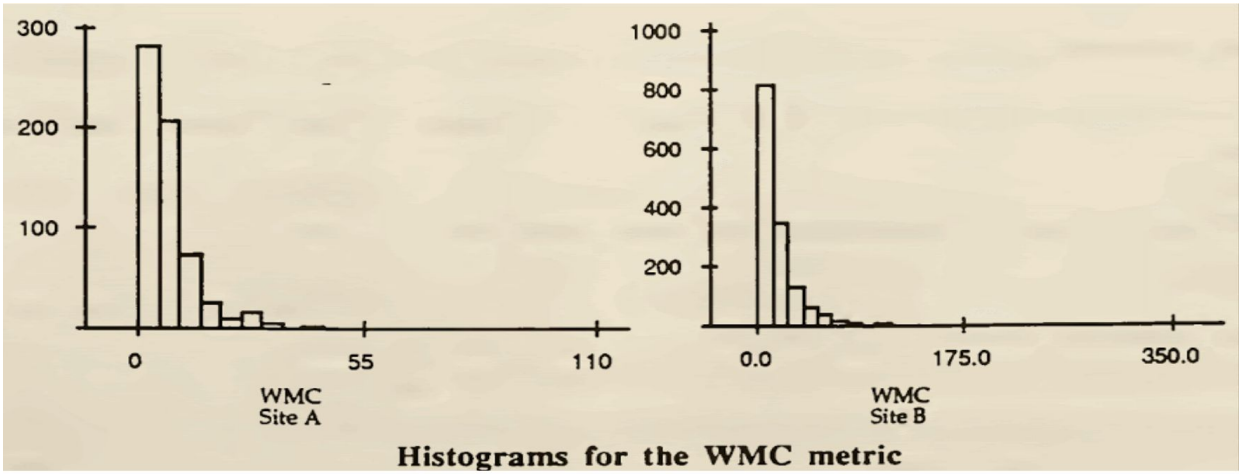
La clase es el bloque constructivo principal de un sistema OO. La clase encapsula operaciones y atributos, y mantienen relaciones con otras clases (herencia, agregación, composición, uso)

- Métricas Chidamber y Kemerer (1994)
  - Aplicadas independientemente del lenguaje de prog. en uso
- Métricas Lorenz y Kidd
- Métricas orientadas a operaciones
- Métricas para pruebas (Binder 1994)
- Métricas para proyectos OO

Métricas CK (Chidamber y Kemerer - 1994)

1) Métodos Ponderados por Clases (MPC) (*Weighted Methods per Class **WMC***)

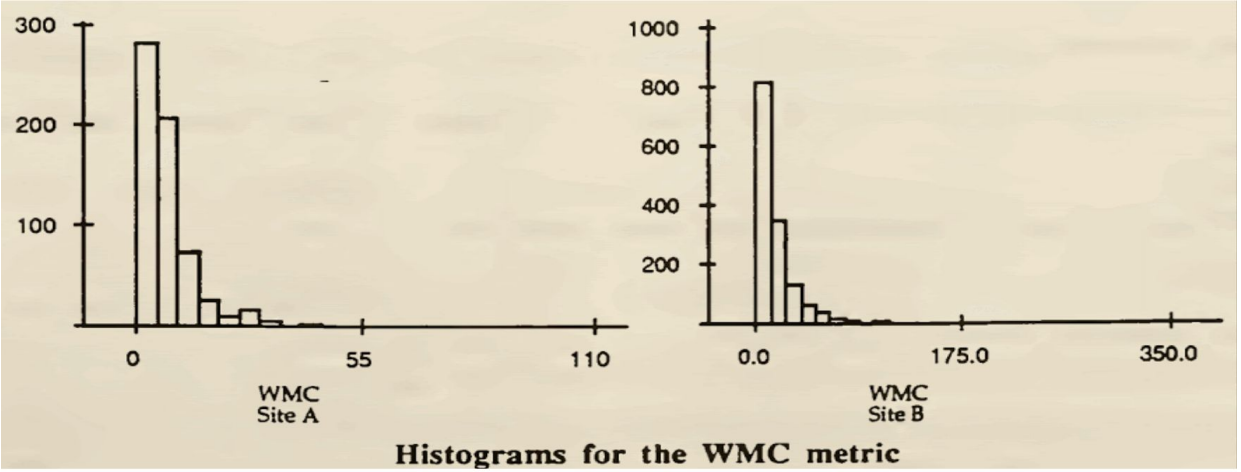
$$WMC = \sum_{1}^{n} c_i$$



- El número y la complejidad de los métodos es un predictor del esfuerzo requerido para el desarrollo y mantenimiento de la clase
- Más grande el número de métodos,
  - más alto el impacto en las clases derivadas.
  - más limitada la posibilidad de reuso (es probable que la clase posea métodos específicos de la aplicación)

Métricas CK (Chidamber y Kemerer - 1994)

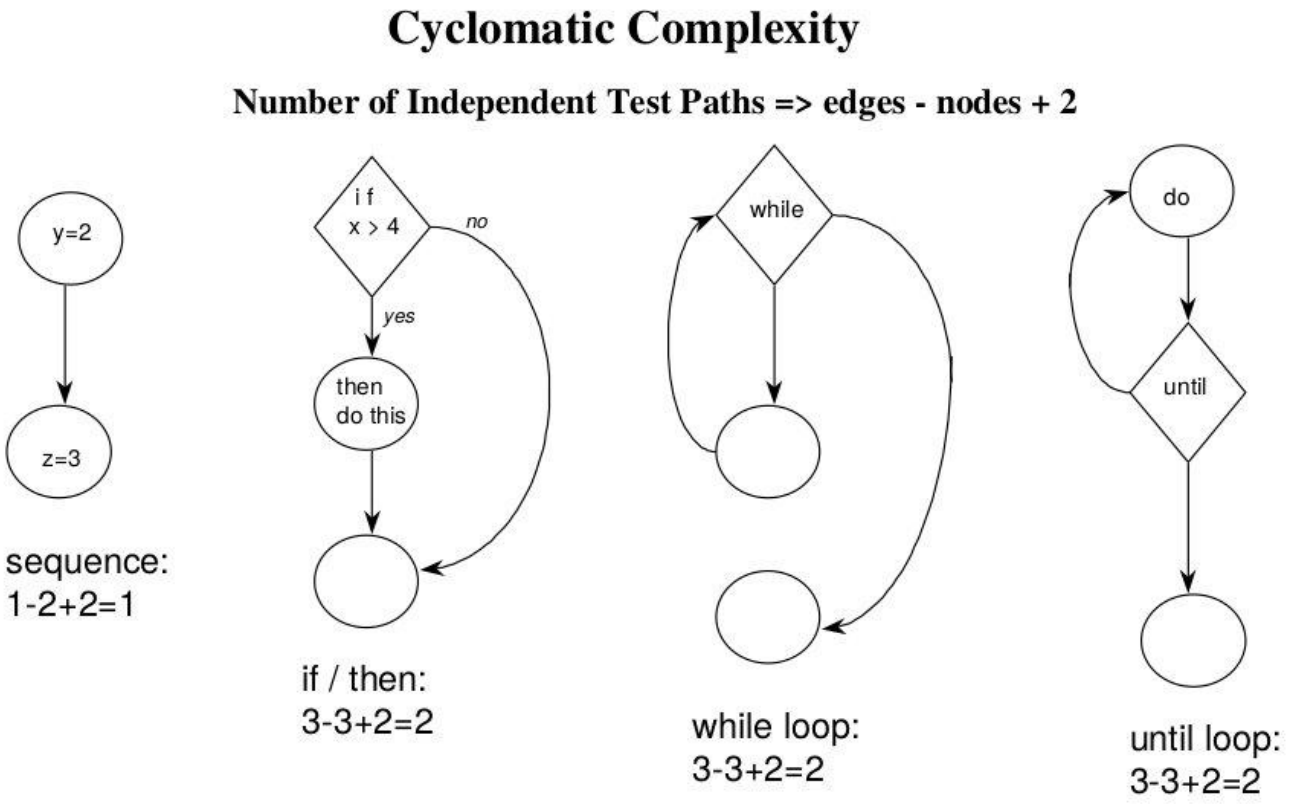
1) Métodos Ponderados por Clases (MPC) (*Weighted Methods per Class **WMC***)



Histograms for the WMC metric

Site	Metric	Mean	Median	Max	Min	StdDev	Skewness
A	WMC	7.69	5	106	0	10.15	4.38
B	WMC	16.39	10	346	0	21.61	4.96

Métrica de complejidad



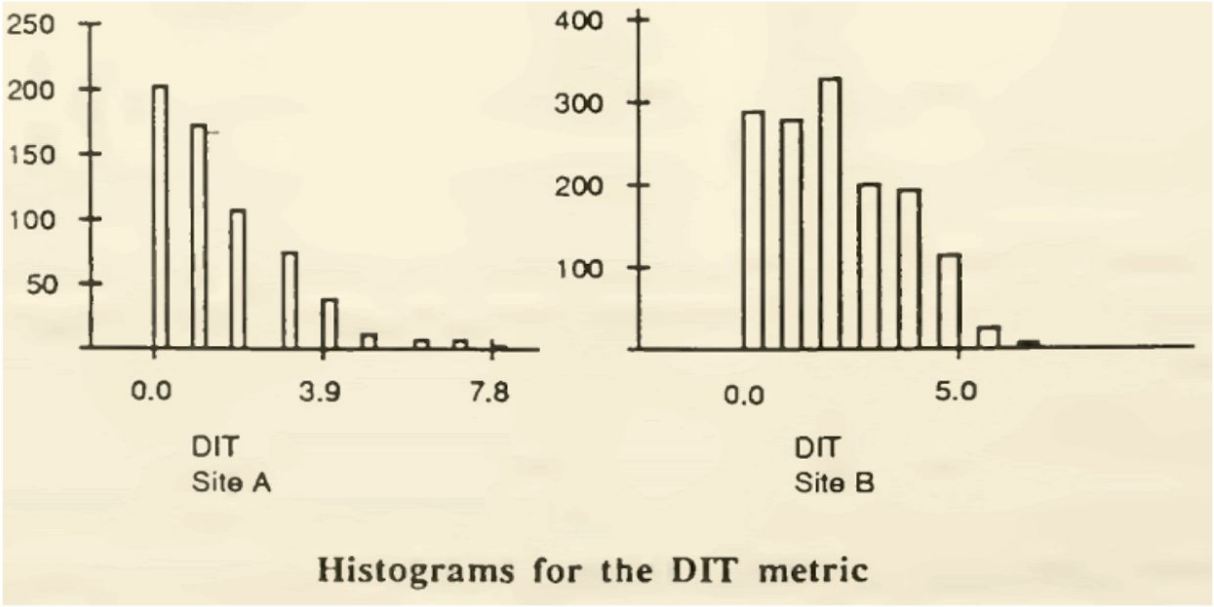
[Rosenberg, Applying and Interpreting Object Oriented Metrics, 1998]

- Podría usarse para la ponderación de la complejidad de los métodos en el cálculo de WMC

Métricas Chidamber y Kemerer (1994)

2) Profundidad del Árbol de Herencia (Depth of Inheritance Tree **DIT**)

La métrica DIT para una clase es la profundidad en el árbol. En herencia múltiple se toma el máximo.

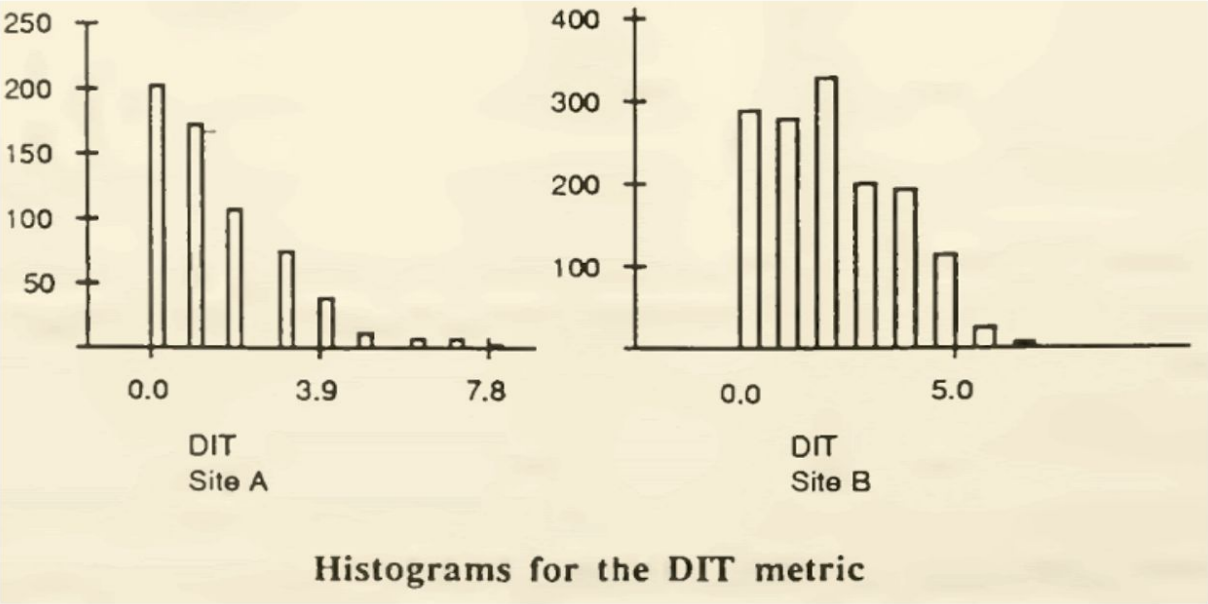


Histograms for the DIT metric

- Más profundo en la jerarquía, más métodos probablemente hereda, lo que la hace más compleja de predecir su comportamiento.
- Árboles profundos representan una complejidad de diseño mayor porque más métodos y clases están involucrados
- Cuando más profundo una clase se halla en la jerarquía, mayor el potencial de uso de métodos heredados.

Métricas Chidamber y Kemerer (1994)

2) Profundidad del Árbol de Herencia (Depth of Inheritance Tree **DIT**)



Histograms for the DIT metric

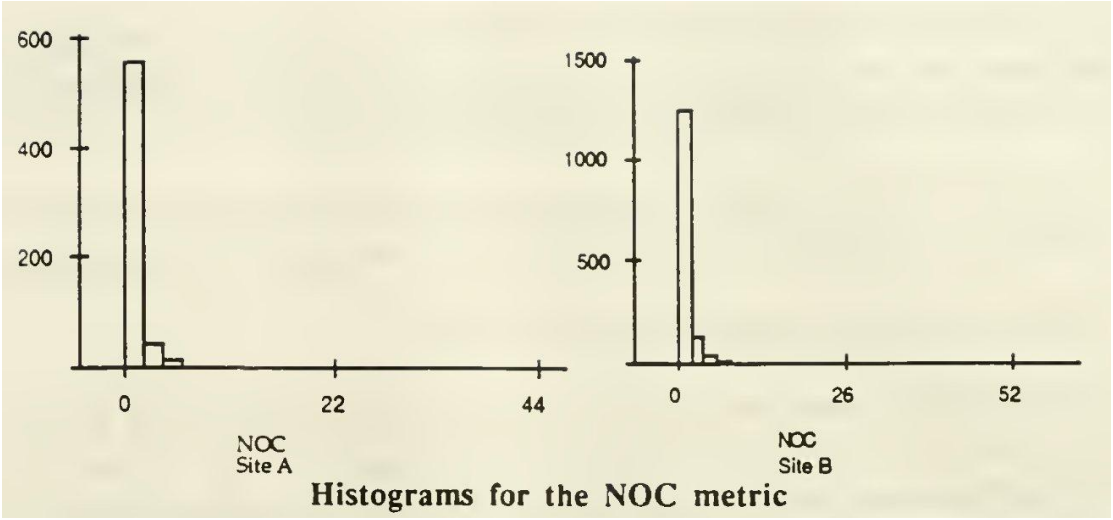
Site	Metric	Mean	Median	Max	Min	StdDev	Skewness
A	DIT	1.54	1	8	0	1.63	1.32
B	DIT	3.16	3	10	0	1.70	0.53

Métricas Chidamber y Kemerer (1994)

3) Número de Hijos (*Number of Children* ***NOC***)

Número de descendiente (NDD)

Número de subclases inmediatas subordinadas a la clase en la jerarquía de clase

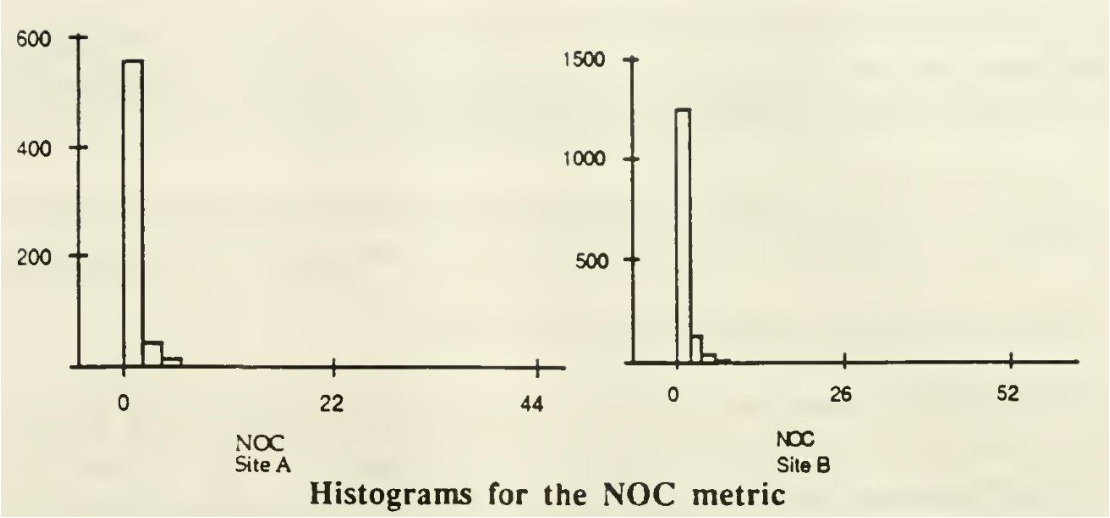


Histograms for the NOC metric

- Más número de subclases
  - más reutilización ya que la herencia es una forma de reuso de métodos heredados.
  - más probabilidad de una abstracción impropia en la clase base. Si una clase tiene un gran número de hijos, puede ser un caso de uso incorrecto de herencia.
- El número de hijos da una idea de la influencia que una clase tiene en el diseño. Más número de hijos, más testing

Métricas Chidamber y Kemerer (1994)

3) Número de Hijos (*Number of Children* ***NOC***)



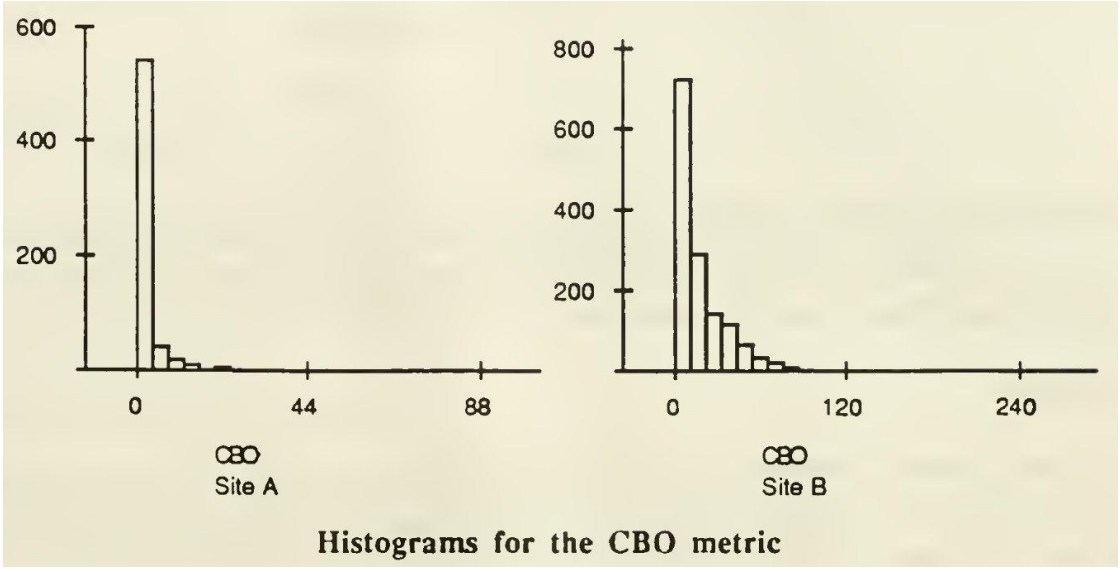
Histograms for the NOC metric

Site	Metric	Mean	Median	Max	Min	StdDev	Skewness
A	NOC	0.67	0	42	0	2.61	9.11
B	NOC	0.66	0	50	0	2.41	12.55

Métricas Chidamber y Kemerer (1994)

4) Acoplamiento entre Objetos (ACO) (*Coupling Between Object classes* ***CBO***)

Dos clases están acopladas cuando los métodos declarados en una clase usa métodos o variables de instancia de otra clase (ambas direcciones)

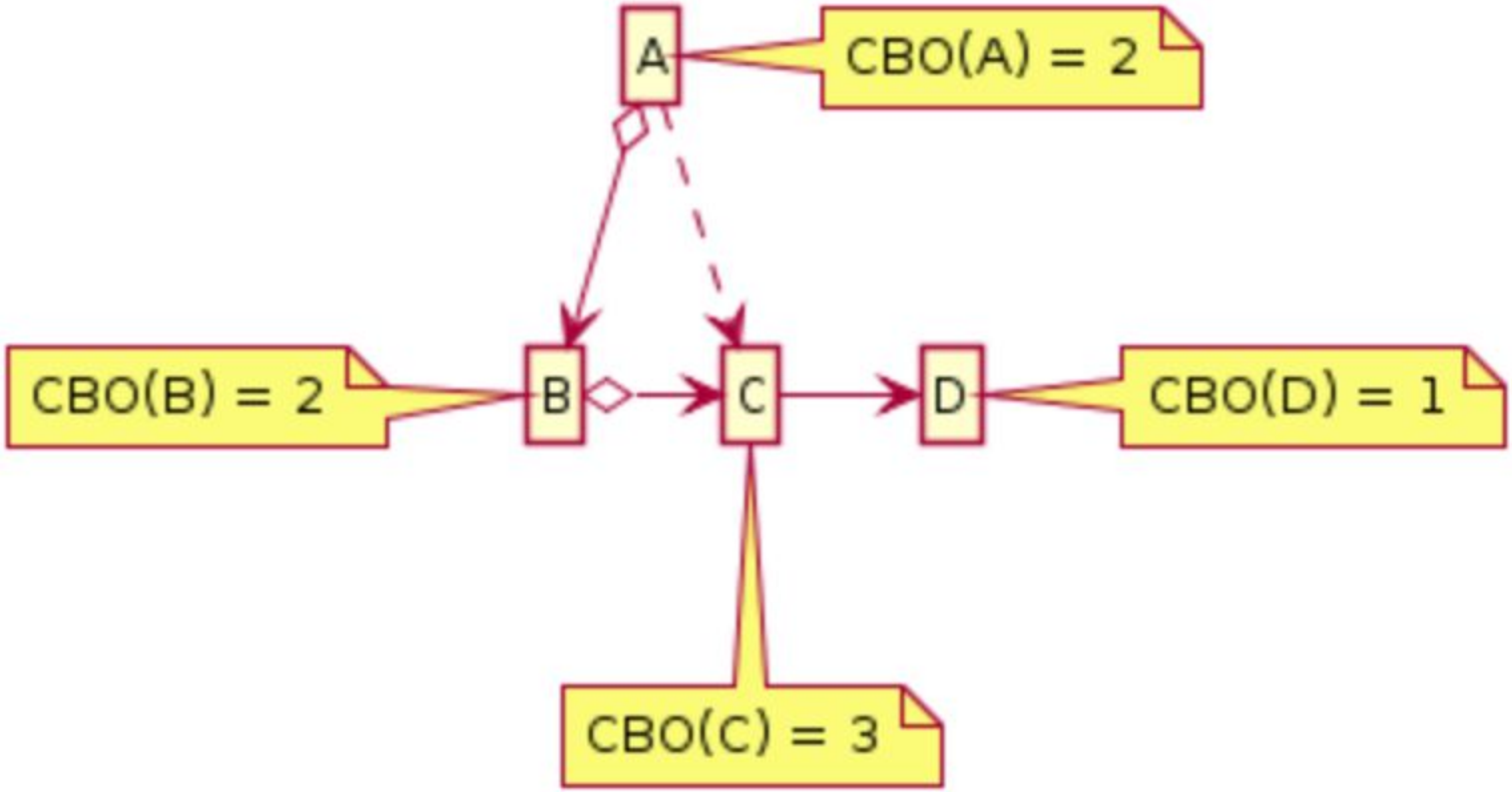


Histograms for the CBO metric

- El acoplamiento excesivo entre clases de objetos es perjudicial para el diseño modular y obstaculiza el reuso. Más independiente una clase, más fácil de reusarla y testearla.
- El acoplamiento entre clases debe ser mantenido al mínimo para favorecer el mantenimiento.
- Un alto acoplamiento determina un testeo complejo para la clase.

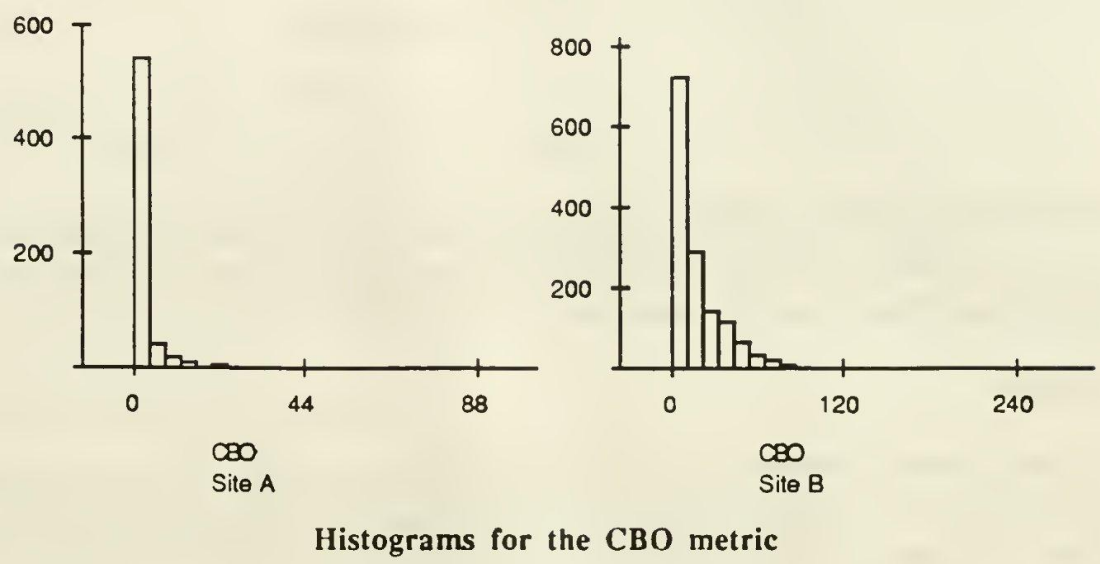
Métricas Chidamber y Kemerer (1994)

4) Acoplamiento entre Objetos (ACO) (*Coupling Between Object classes* ***CBO***)



Métricas Chidamber y Kemerer (1994)

4) Acoplamiento entre Objetos (ACO) (*Coupling Between Object classes **CBO***)



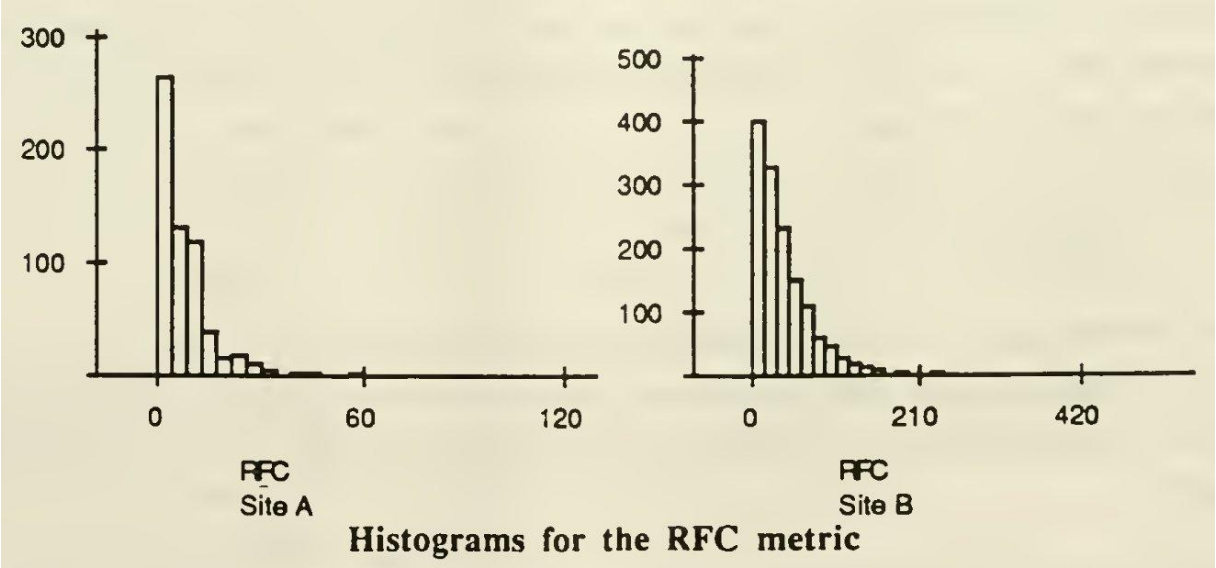
Histograms for the CBO metric

Site	Metric	Mean	Median	Max	Min	StdDev	Skewness
A	CBO	2.01	0	84	0	5.56	7.25
B	CBO	16.90	9	234	0	22.21	3.04

Métricas Chidamber y Kemerer (1994)

5) Respuesta Para una Clase (RPC) (*Response For a class **RFC***)

El conjunto de respuesta de una clase es el conjunto de métodos que potencialmente pueden ser ejecutados en respuesta a un mensaje recibido por un objeto de la clase.

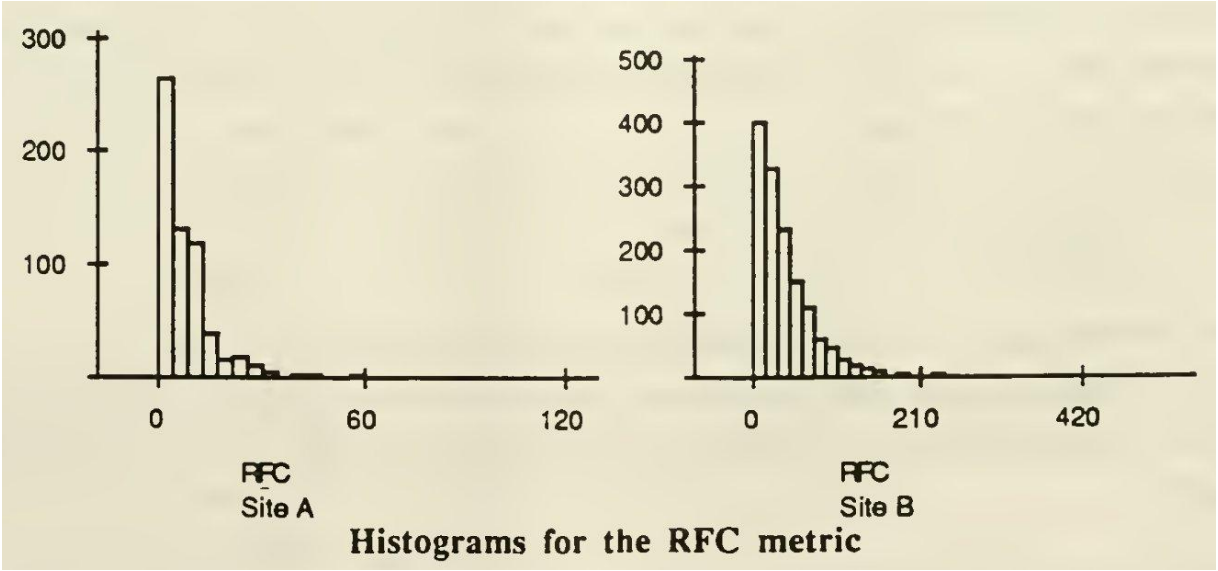


Histograms for the RFC metric

- Cuando más grande el número de métodos que pueden ser invocados en respuesta a un mensaje:
  - el testing y debugging se vuelve complejo
  - la clase es más compleja
- Un peor caso de esta métrica puede ser un indicio del tiempo de testing necesario

Métricas Chidamber y Kemerer (1994)

5) Respuesta Para una Clase (RPC) (*Response For a class **RFC***)



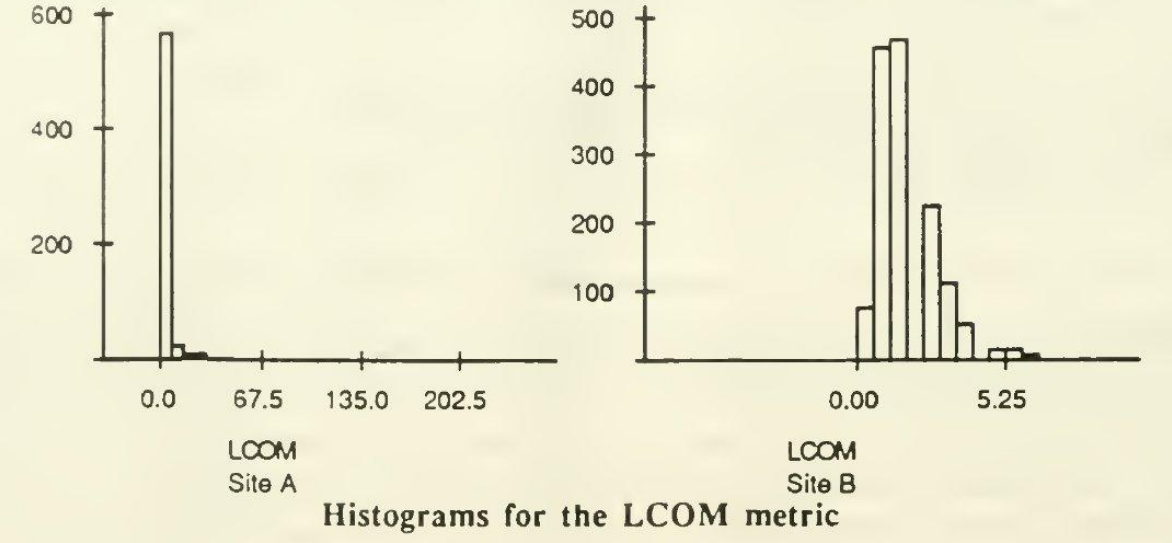
Histograms for the RFC metric

Site	Metric	Mean	Median	Max	Min	StdDev	Skewness
A	RFC	10.00	6	120	0	13.78	3.70
B	RFC	42.35	29	422	3	45.24	2.95

Métricas Chidamber y Kemerer (1994)

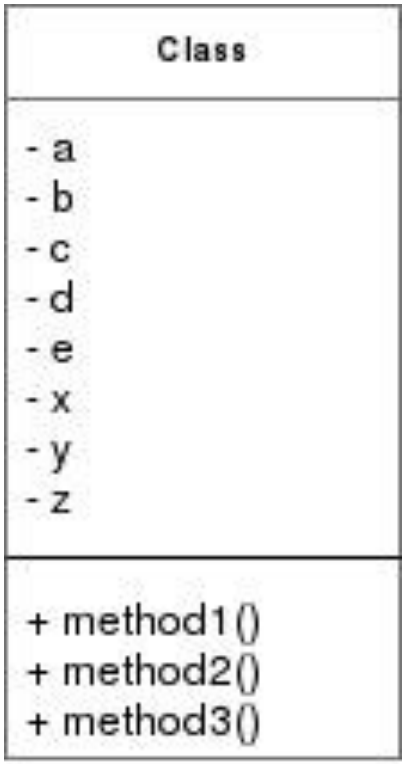
6) Carencia de Cohesión en Métodos (*Lack of Cohesion in Methods* ***LCOM***)

Se basa en el grado de similitud de los métodos. Es la suma de los pares de métodos con similitud cero (conjunto nulo) menos la suma de los pares de métodos con similitud más que cero. La similitud se mide por acceso a las variables de instancias

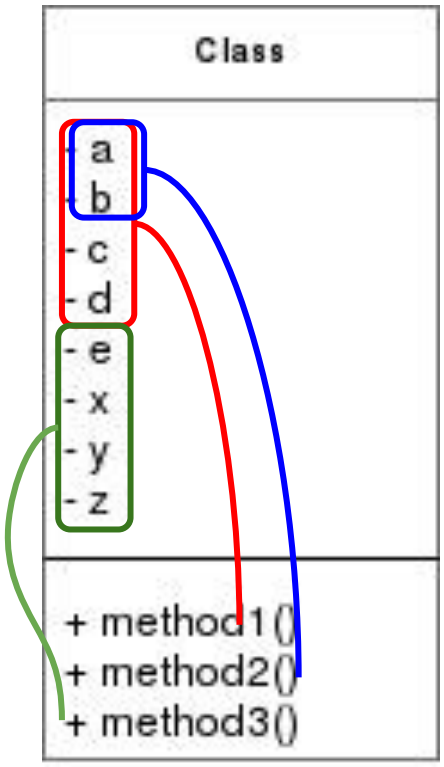


- Se usa para medir cuan cohesiva es una clase o, en contraposición, su disparidad funcional (Altos valores puede interpretarse como una ‘God class’)
- Se desea métodos cohesivos ya que promueven la encapsulación.
- Falta de cohesión indicaría que la clase deba ser separada en dos o más clases
- Una alta disparidad funcional señala problemas en el diseño de clases
- Una baja cohesión incrementa la complejidad y el costo de desarrollo y mantenimiento.

Ejemplo



Ejemplo



- metodo1 accede {a, b, c, d}
- metodo2 accede {a, b}
- metodo3 accede {e, x, y, z}
- El conjunto de variables de instancias accedidas por los métodos determinan:
  - 1 intersección no nula
  - 2 intersecciones nulas
- LCOM resulta

|P| = num intersecciones nulas

|Q| = num intersecciones no nulas

LCOM = |P| - |Q|, if |P| > |Q|

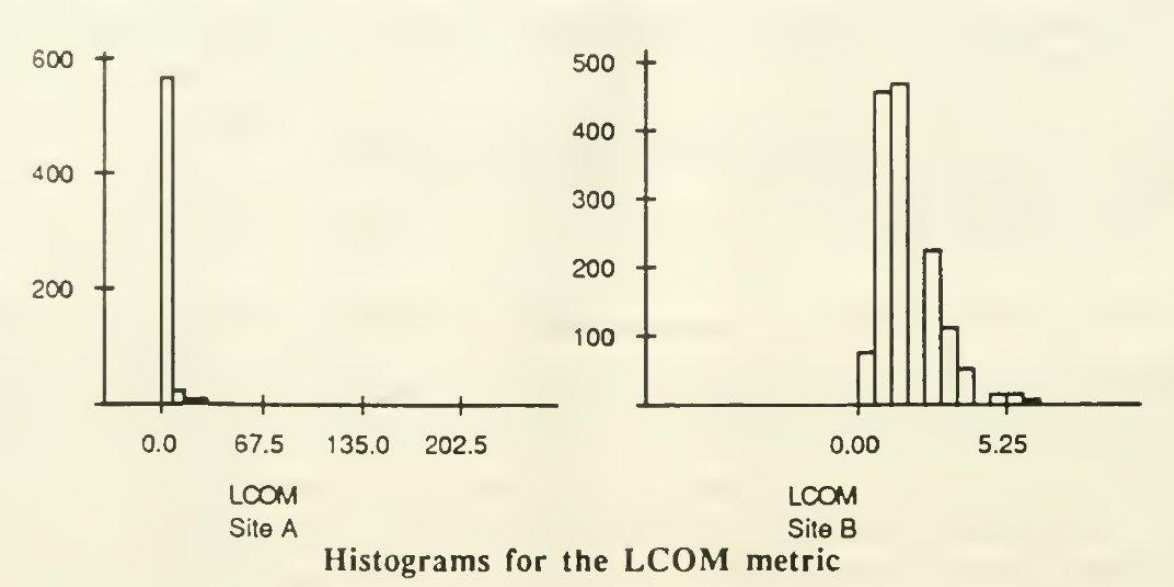
= 0 , de otra forma

LCOM = 2 - 1 = 1

Buscamos una métrica LCOM con valores altos o bajos?

Métricas Chidamber y Kemerer (1994)

6) Carencia de Cohesión en Métodos (*Lack of Cohesion in Methods* ***LCOM***)



Histograms for the LCOM metric

Site	Metric	Mean	Median	Max	Min	StdDev	Skewness
A	LCOM	4.03	0	200	0	16.94	7.31
B	LCOM	2.23	2	17	0	1.70	2.54

Métricas Lorenz y Kidd (1994)

Dividen las métricas en cuatro grandes categorías:

- **Tamaño**: centradas en recuentos de atributos y operaciones para una clase individual y promedian los valores para el sistema OO en su totalidad.
- **Herencia**: centradas en cómo se reutilizan las operaciones a lo largo y ancho de la jerarquía de clases.
- **Valores Internos**: examinan la cohesión
- **Valores externos**: examinan el acoplamiento y la reutilización.

Métricas Lorenz y Kidd (1994)

Tamaño de Clase *(Class Size - CS)*:

- Número total de operaciones (heredadas + privadas)
- Número de atributos (heredados + privados)

- Un CS grande indica una clase con mucha responsabilidad lo que reduce la reusabilidad y complica la implementación y testeo.
- Atributos y operaciones heredadas o públicas deben ser ponderadas más pesadamente para establecer el tamaño.
- Atributos y operaciones privados permiten la especialización y están más localizados en el diseño.
- Promedios de atributos y operaciones bajos favorecen la reutilización

Métricas Lorenz y Kidd (1994)

Número de Operaciones Invalidadas por una subclase

(Number of Operations Overridden (NOO) by a Subclass):

- Se invalida una operación de una clase heredada al sustituir la implementación de la misma
- Un valor grande del NOO indica un problema de diseño
  - Se violó la abstracción que representa la superclase
  - La subclase debería limitarse a extender los servicios de la superclase
  - Da lugar a una jerarquía de clases débil y a una aplicación difícil de testear y modificar

Métricas Lorenz y Kidd (1994)

Número de Operaciones Añadidas por una subclase (Number of Operations Added (NOA) by a Subclass):

- Las subclases se especializan con adición de operaciones y atributos privados.
- Cuando crece la profundidad de la jerarquía de clases, el nivel NOA en niveles inferiores debería disminuir

Métricas Lorenz y Kidd (1994)

Índice de Especialización (Specialization Index (SI)):

- Indica aproximadamente el grado de especialización de una subclase existente en un sistema OO
- La especialización se puede lograr con:
  - Agregar una nueva operación
  - Sobreescribir una operación existente de clase base

$$SI = \frac{NOO * nivel\ jerarquia}{num\ total\ de\ metodos}$$

- Ayuda a evaluar la calidad de una subclase
- Una buena subclase es generalmente una extensión de las capacidades de su clase base
- Una valor alto puede indicar un diseño de subclase pobre

Métricas Orientadas a Operaciones (L&K

contenido opcional

La clase es el bloque constructivo fundamental de los sistemas OO, por lo tanto hay más métricas relativas a clases.

- Tamaño Medio de Operación (OS_{avg})
  - se usa SLOC del método o el número de mensajes enviados desde la operación. Si el número crece es probable que las responsabilidades no están bien determinadas.
- Complejidad de Operación (OC)
  - se usa cualquier métrica de complejidad. Como las operaciones están limitadas a brindar sólo una operación, el valor OC debe ser lo más bajo posible.
- Número Medio de Parámetros por Operación (NP_{avg})
  - Cuando más alto es este valor, más compleja la colaboración y puede ser síntoma de la falta de una abstracción. Se busca un valor lo más bajo posible

Métricas para Pruebas OO (Binder 94)

contenido opcional

Se centra en las características importantes del diseño con influencia directa en la testeabilidad del sistema:

- **Testeabilidad y encapsulamiento**
  - Falta de Cohesión en los Métodos (LCOM)
  - Porcentaje público y protegido (PPP)
  - Acceso Público a Datos Miembros (PAD)
- **Testeabilidad y herencia**
  - Número de Clases Raíz (NOR)
  - Admisión (*Fan-in (FIN))*
  - Número de Hijos (NOC)
  - Profundidad del Árbol de herencia (DIT)

Métricas para Pruebas OO (Binder 94)

contenido opcional

Testeabilidad y encapsulamiento

- Falta de Cohesión en los Métodos (LCOM)
  - Cuando más alto es LCOM, más estado hay que testear para asegurar que lo métodos no den a lugar a efectos laterales
- Porcentaje público y protegido (PPP)
  - Los atributos públicos son heredados desde otras clases y son visibles en esas clases.
  - Los atributos protegidos son accesibles en los métodos de subclases.
  - El porcentaje de atributos que son públicos o protegidos indica la proporción de información visible desde otros objetos.
  - Un alto valor incrementa la probabilidad de efecto lateral entre clases porque los atributos públicos y protegidos conllevan a un potencial acoplamiento.
  - El testeo trata de descubrir estos side effects.

Métricas para Pruebas OO (Binder 94)

contenido opcional

Testeabilidad y encapsulamiento

- Acceso Público a Datos Miembros (PAD)
  - Número de clases que pueden acceder a datos miembros de otras clases (públicos o protegidos)
  - Es una medida de la ruptura del encapsulamiento
  - Valores altos indican una alta probabilidad de efectos laterales lo que genera un mayor esfuerzo de test para descubrir estos side effects.

Métricas para Pruebas OO (Binder 94)

contenido opcional

Testeabilidad y herencia

- Número de Clases Raíz (NOR)
  - Cuenta las distintas jerarquías de clases
  - Se necesita una suite de test diferente para cada jerarquía
  - Si NOR crece, el esfuerzo de test también.
- Admisión (*Fan-in (FIN)*)
  - El número de clases de las que una clase deriva.
  - Cuando hay múltiple herencia FIN > 1, indica que la clase hereda de más de una clase. (No más de 1 en lo posible)
- NOC
  - El número de clases derivadas de un padre específico.
  - Un cambio en una clase base impacta en las subclases.
- DIT
  - Todos los métods heredados tienen que ser testeados en la clase bajo test

Métricas para proyectos (Lorenz y Kidd 9

contenido opcional

Nos dan una idea del esfuerzo requerido para desarrollar el software. Sirven para planificar, coordinar, seguir y controlar un proyecto de software.

- Números de guiones de escenarios
  - directamente proporcional al número de clases
- Número de clases claves
  - Clases que se centran en el dominio del problema.
- Número promedio de clases de apoyo por clase clave
  - Ayudan a la estimación.
- Número de subsistemas
  - Medida del esfuerzo de integración.

Bibliografía

Rosenberg, Applying and Interpreting Object Oriented Metrics, 1998

Chidamber, Shyam R., and Chris F. Kemerer. "A metrics suite for object oriented design." IEEE Transactions on software engineering 20.6 (1994): 476-493.

Binder, Robert V. "Design for testability in object-oriented systems." Communications of the ACM 37.9 (1994): 87-101.

Mark Lorenz, Jeff Kidd, “Object-Oriented Software Metrics”, Prentice Hall, 1994.

Pressman, Roger S. Software engineering: a practitioner's approach. 7th edition. Palgrave Macmillan, (2010).