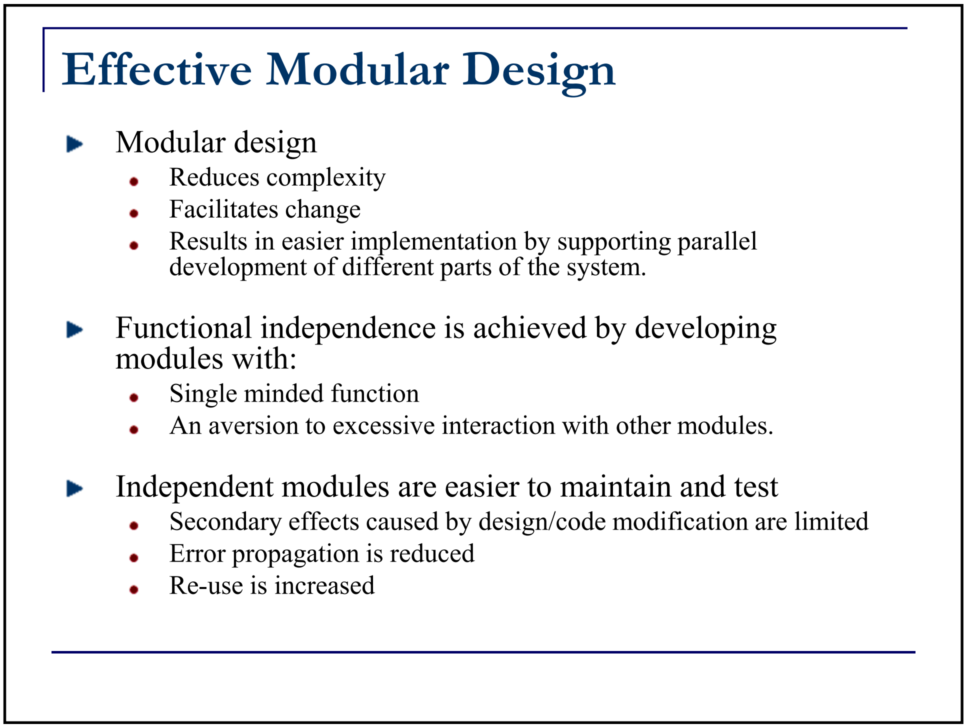
CSC40232: SOFTWARE ENGINEERING

Professor: Jane Cleland-Huang

Metrics

sarec.nd.edu/courses/SE2017

Department of Computer Science and Engineering

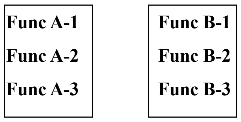


Effective Modular Design

- Modular design
  - Reduces complexity
  - Facilitates change
  - Results in easier implementation by supporting parallel development of different parts of the system.
- Functional independence is achieved by developing modules with:
  - Single minded function
  - An aversion to excessive interaction with other modules.
- Independent modules are easier to maintain and test
  - Secondary effects caused by design/code modification are limited
  - Error propagation is reduced
  - Re-use is increased

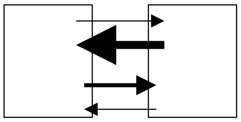
Two qualitative criteria

- **Cohesion**
  A measure of the relative functional strength of a module



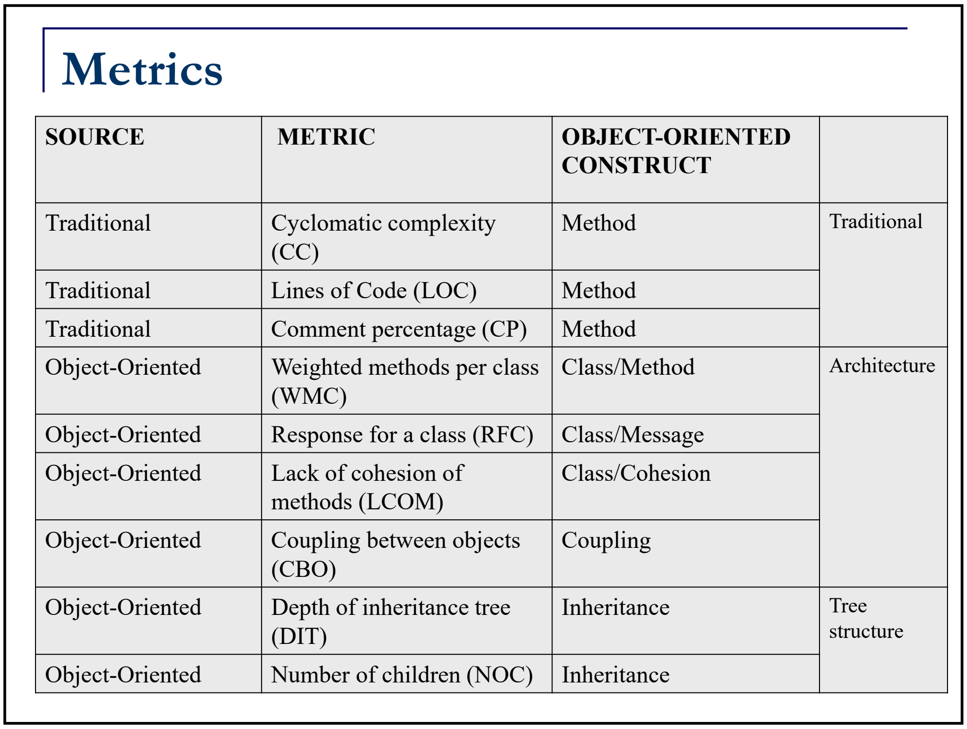
High Cohesion (good)

- **Coupling**
  A measure of the relative interdependence among modules.



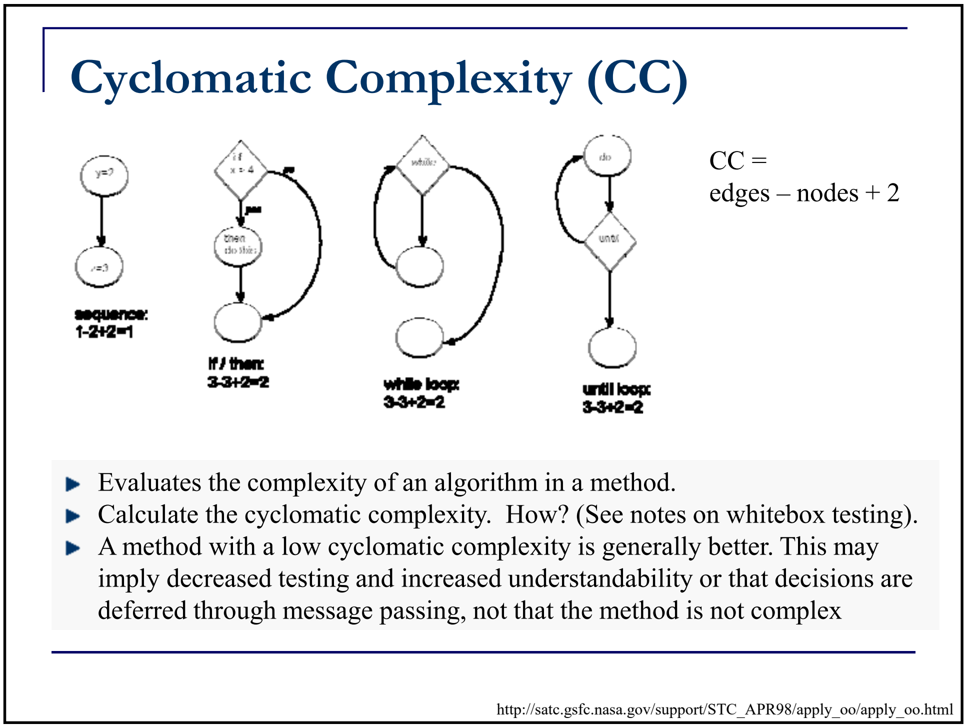
High coupling (bad)

I WANT TO KNOW ABOUT SOFTWARE COMPLEXITY I WANT SOME REAL ANSWERS...
EIGHT
EIGHT ?!
I THOUGHT THAT WAS A PRETTY GOOD ANSWER?



Metrics

SOURCE	METRIC	OBJECT-ORIENTED CONSTRUCT	
Traditional	Cyclomatic complexity (CC)	Method	Traditional
Traditional	Lines of Code (LOC)	Method	
Traditional	Comment percentage (CP)	Method	
Object-Oriented	Weighted methods per class (WMC)	Class/Method	Architecture
Object-Oriented	Response for a class (RFC)	Class/Message	
Object-Oriented	Lack of cohesion of methods (LCOM)	Class/Cohesion	
Object-Oriented	Coupling between objects (CBO)	Coupling	
Object-Oriented	Depth of inheritance tree (DIT)	Inheritance	Tree structure
Object-Oriented	Number of children (NOC)	Inheritance	

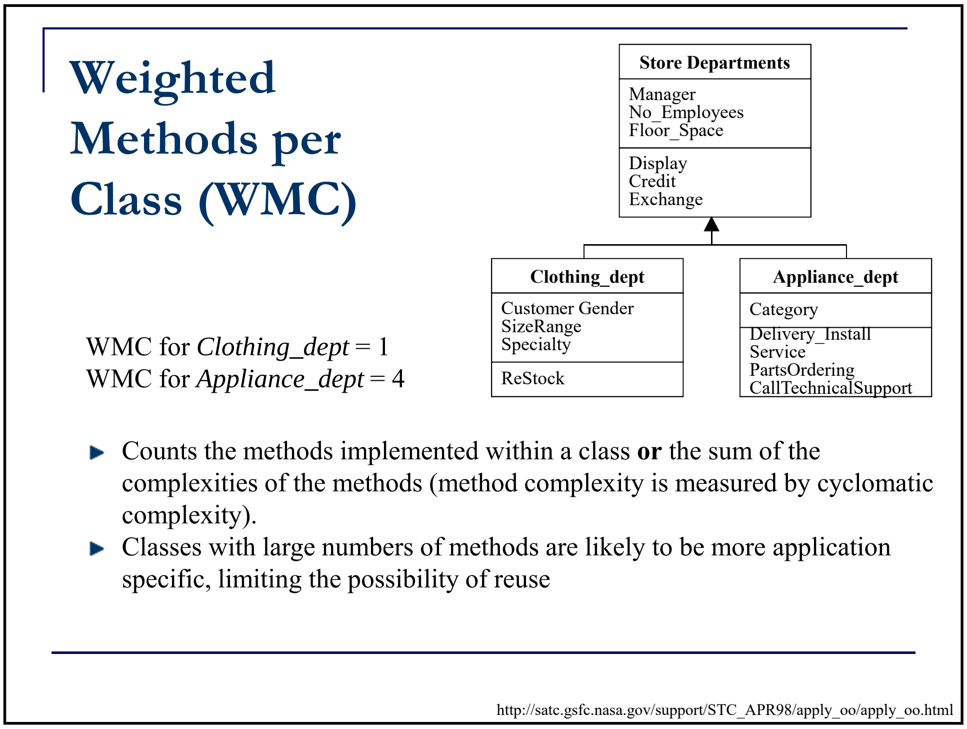


Cyclomatic Complexity (CC)

CC =
edges – nodes + 2

- Evaluates the complexity of an algorithm in a method.
- Calculate the cyclomatic complexity. How? (See notes on whitebox testing).
- A method with a low cyclomatic complexity is generally better. This may imply decreased testing and increased understandability or that decisions are deferred through message passing, not that the method is not complex

http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



Weighted Methods per Class (WMC)

Store Departments

Manager
No_Employees
Floor_Space

Display
Credit
Exchange

Clothing_dept

Customer Gender
SizeRange
Specialty

ReStock

Appliance_dept

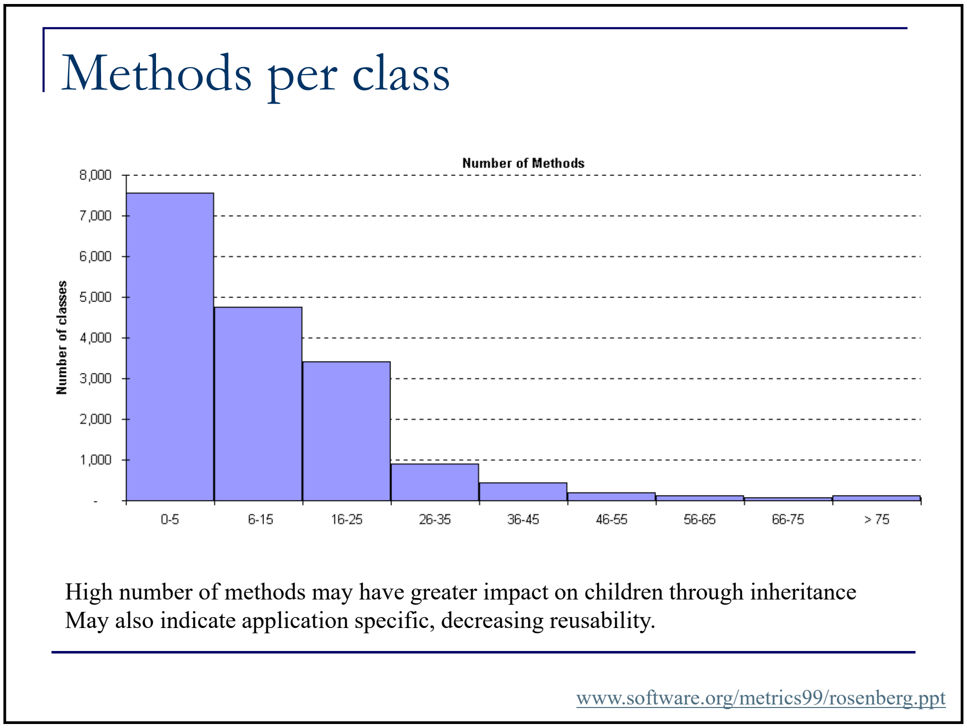
Category

Delivery_Install
Service
PartsOrdering
CallTechnicalSupport

WMC for *Clothing_dept* = 1
WMC for *Appliance_dept* = 4

- Counts the methods implemented within a class **or** the sum of the complexities of the methods (method complexity is measured by cyclomatic complexity).
- Classes with large numbers of methods are likely to be more application specific, limiting the possibility of reuse

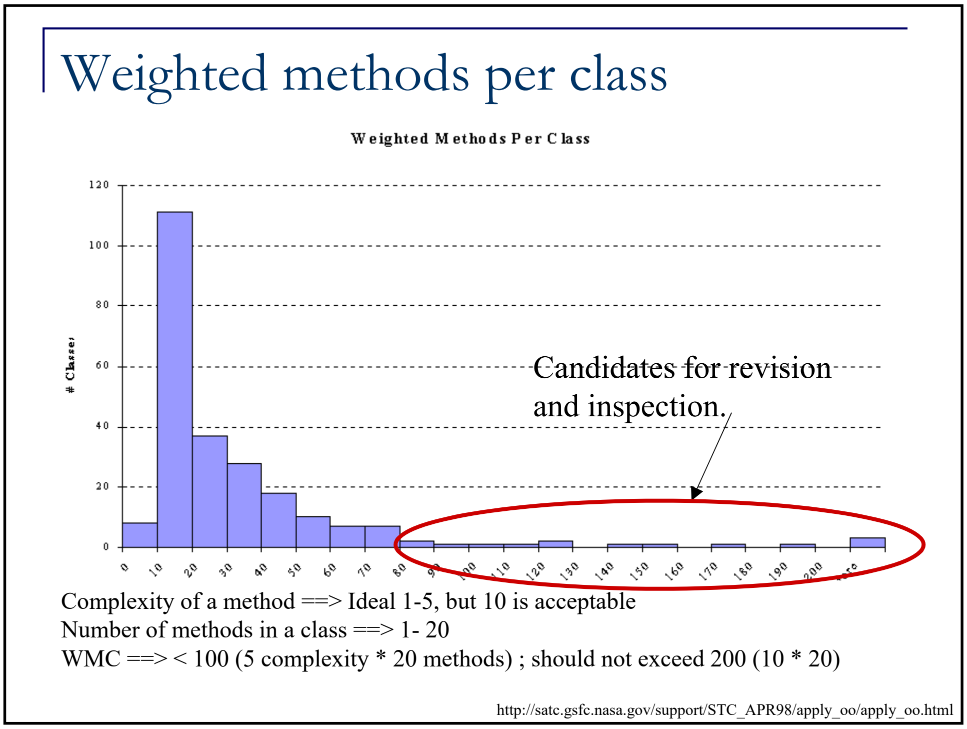
http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



Methods per class

High number of methods may have greater impact on children through inheritance
May also indicate application specific, decreasing reusability.

www.software.org/metrics99/rosenberg.ppt

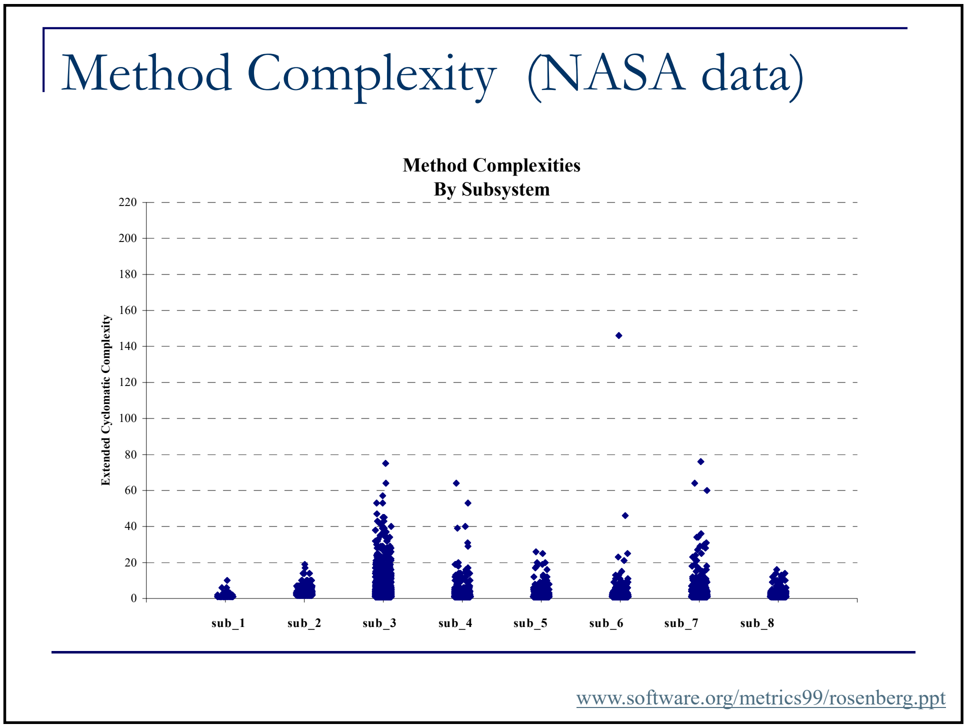


Weighted methods per class

Candidates for revision and inspection.

Complexity of a method ==> Ideal 1-5, but 10 is acceptable
Number of methods in a class ==> 1- 20
WMC ==> < 100 (5 complexity * 20 methods) ; should not exceed 200 (10 * 20)

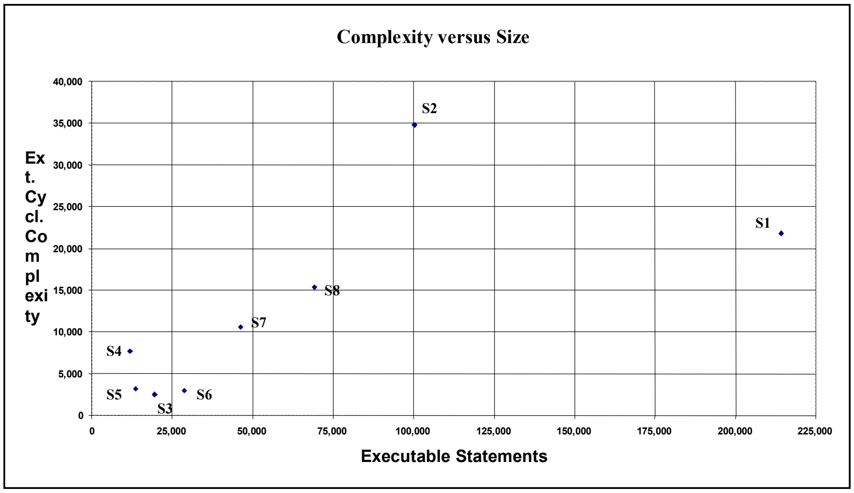
http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



Method Complexity (NASA data)

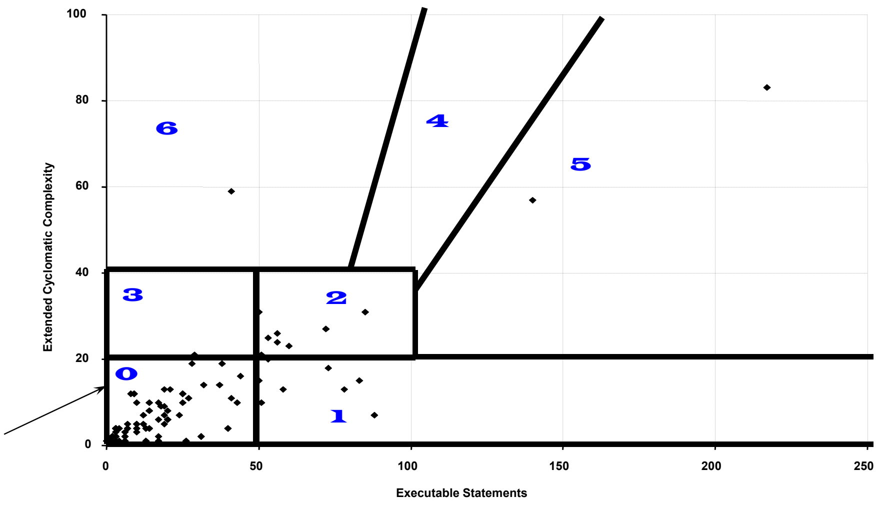
www.software.org/metrics99/rosenberg.ppt

Complexity vs. Size (NASA)

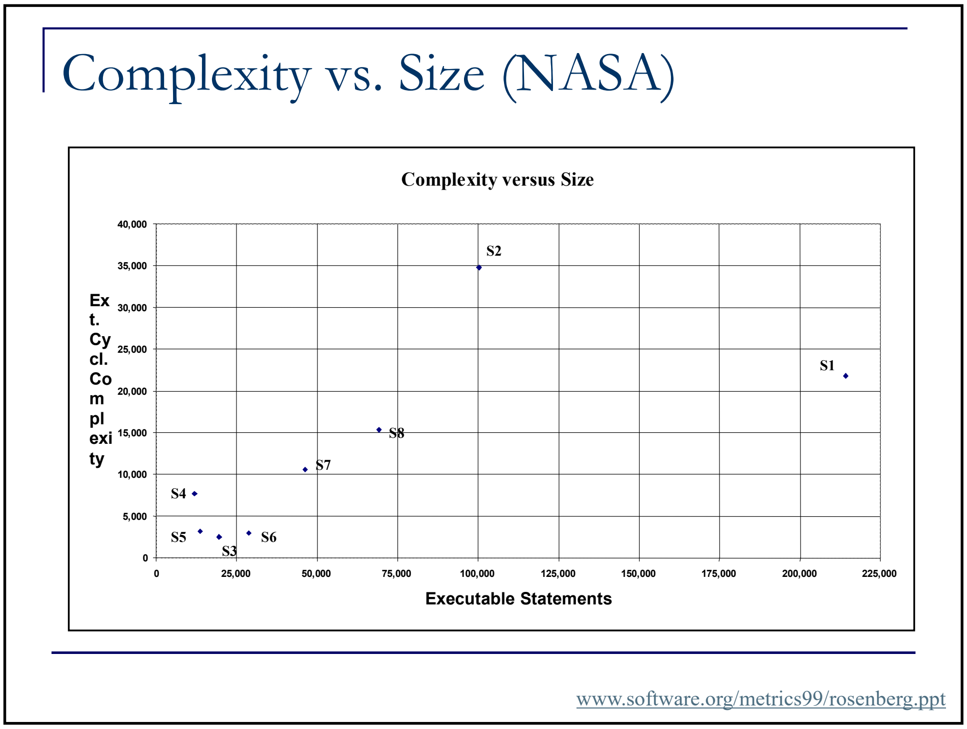


www.software.org/metrics99/rosenberg.ppt

Size to Complexity (RISK components)

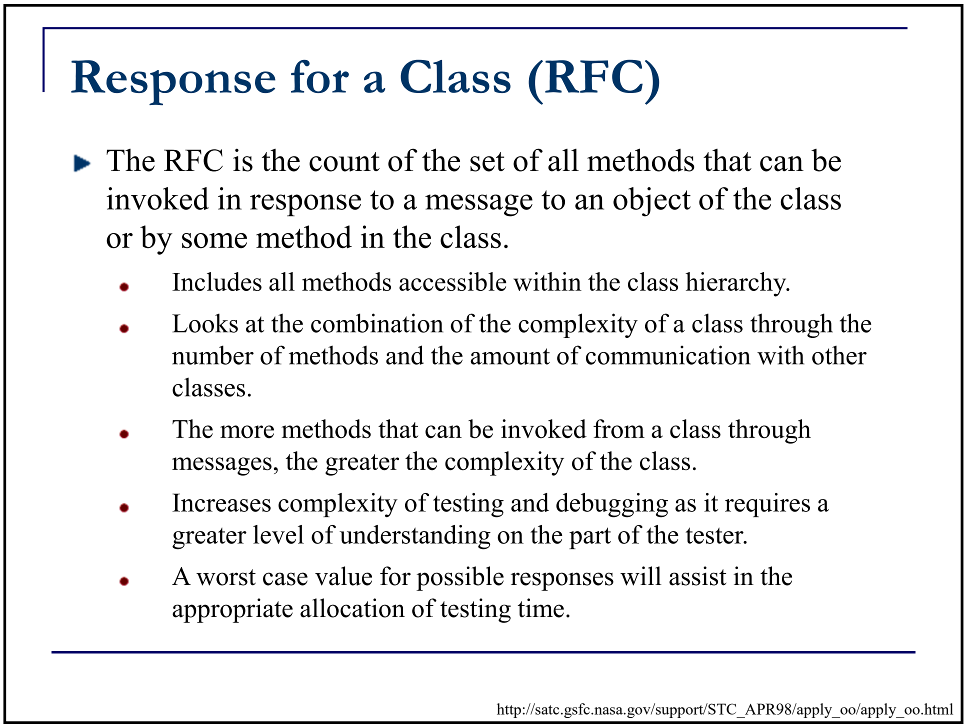


www.software.org/metrics99/rosenberg.ppt



Complexity vs. Size (NASA)

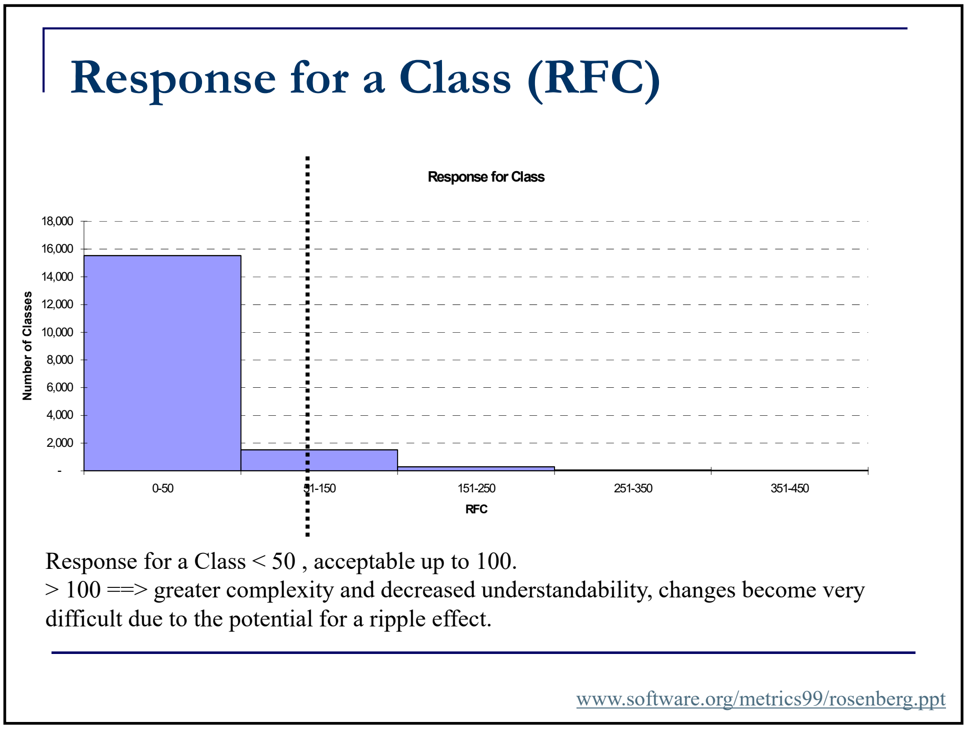
www.software.org/metrics99/rosenberg.ppt



Response for a Class (RFC)

- The RFC is the count of the set of all methods that can be invoked in response to a message to an object of the class or by some method in the class.
  - Includes all methods accessible within the class hierarchy.
  - Looks at the combination of the complexity of a class through the number of methods and the amount of communication with other classes.
  - The more methods that can be invoked from a class through messages, the greater the complexity of the class.
  - Increases complexity of testing and debugging as it requires a greater level of understanding on the part of the tester.
  - A worst case value for possible responses will assist in the appropriate allocation of testing time.

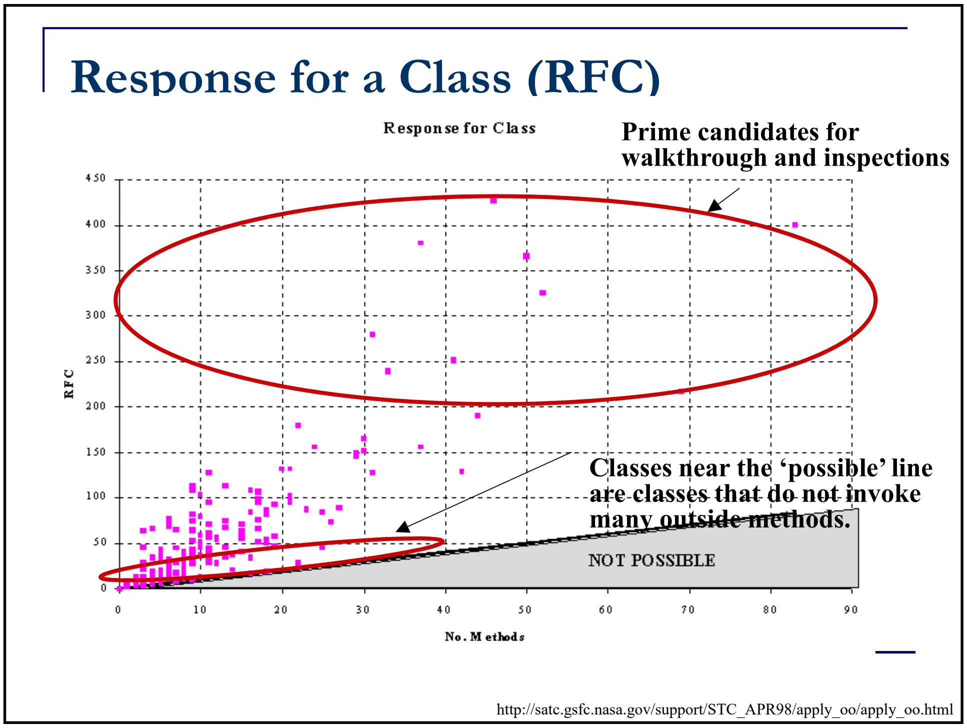
http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



Response for a Class (RFC)

Response for a Class < 50 , acceptable up to 100.
> 100 ==> greater complexity and decreased understandability, changes become very difficult due to the potential for a ripple effect.

www.software.org/metrics99/rosenberg.ppt



Response for a Class (RFC)

Prime candidates for walkthrough and inspections

Classes near the 'possible' line are classes that do not invoke many outside methods.

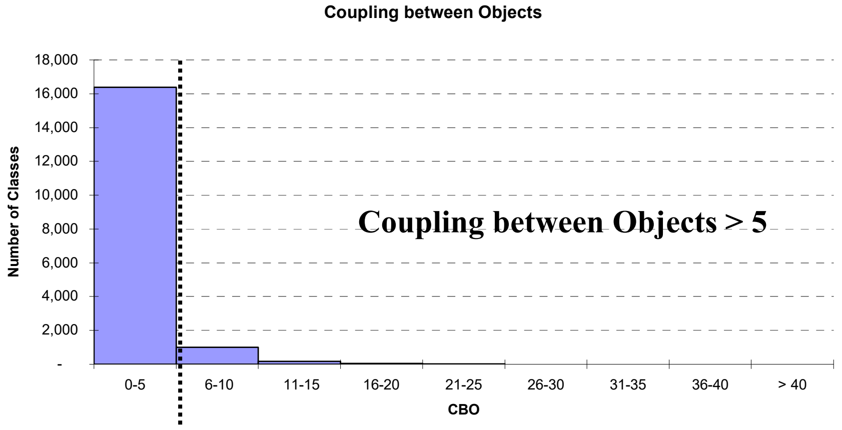
http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html

Coupling between Object Classes (COB)

- Count of the number of other classes to which a class is coupled.
  - Count the number of distinct non-inheritance related class hierarchies on which a class depends.
- Excessive coupling is detrimental to modular design and prevents reuse.
- High COB:
  - Prevents reuse.
  - Increases sensitivity to changes in other parts of the design. Therefore maintenance becomes harder.
  - Understandability decreases.
- Design classes with weak coupling.

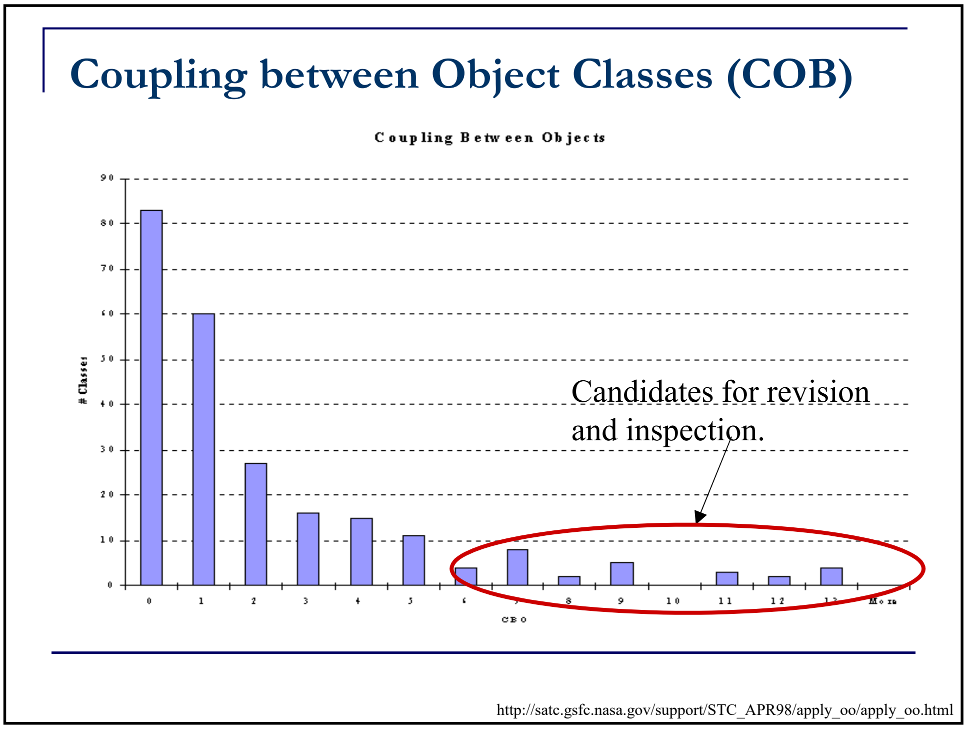
http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html

Coupling between Object Classes (COB)



Higher CBO indicates classes that my be difficult to understand
Decreased reuse and increased maintenance.

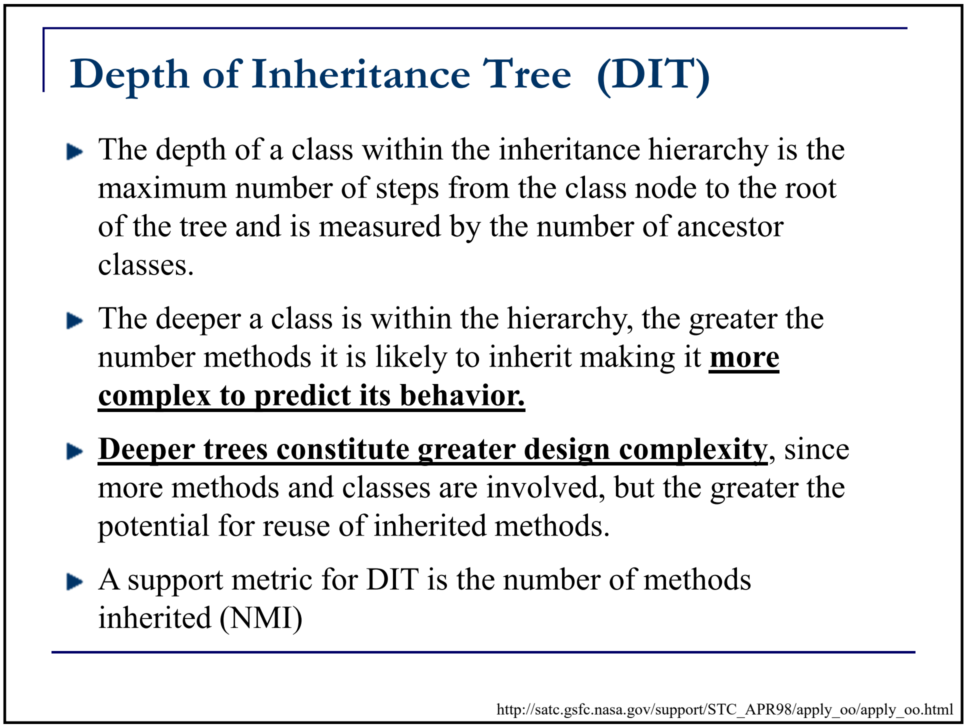
www.software.org/metrics99/rosenberg.ppt



Coupling between Object Classes (COB)

Candidates for revision and inspection.

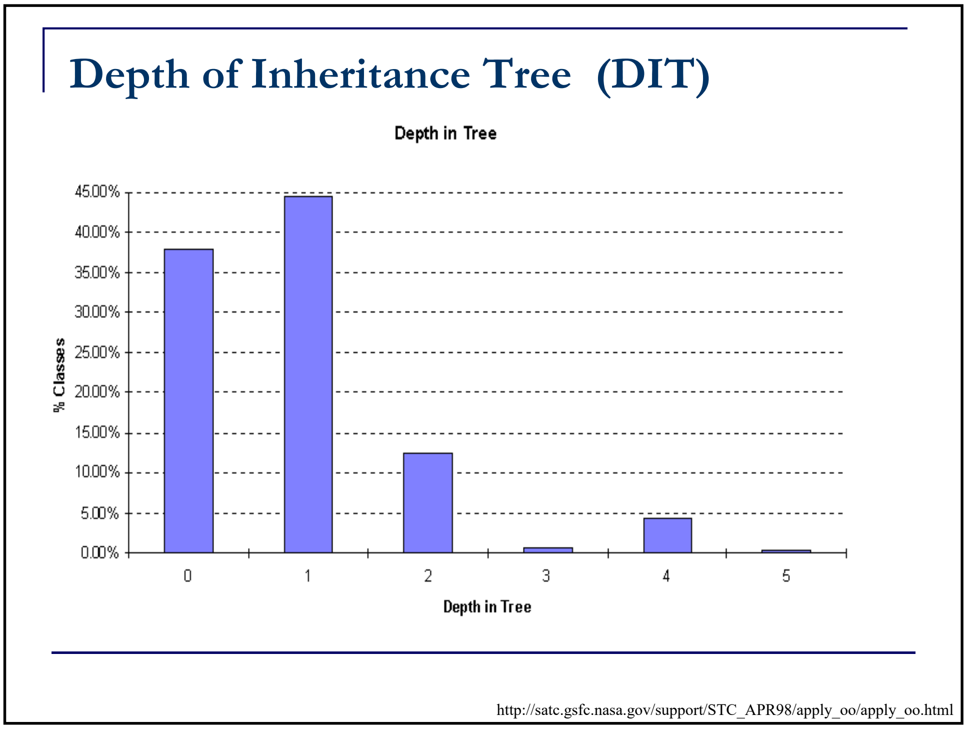
http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



Depth of Inheritance Tree (DIT)

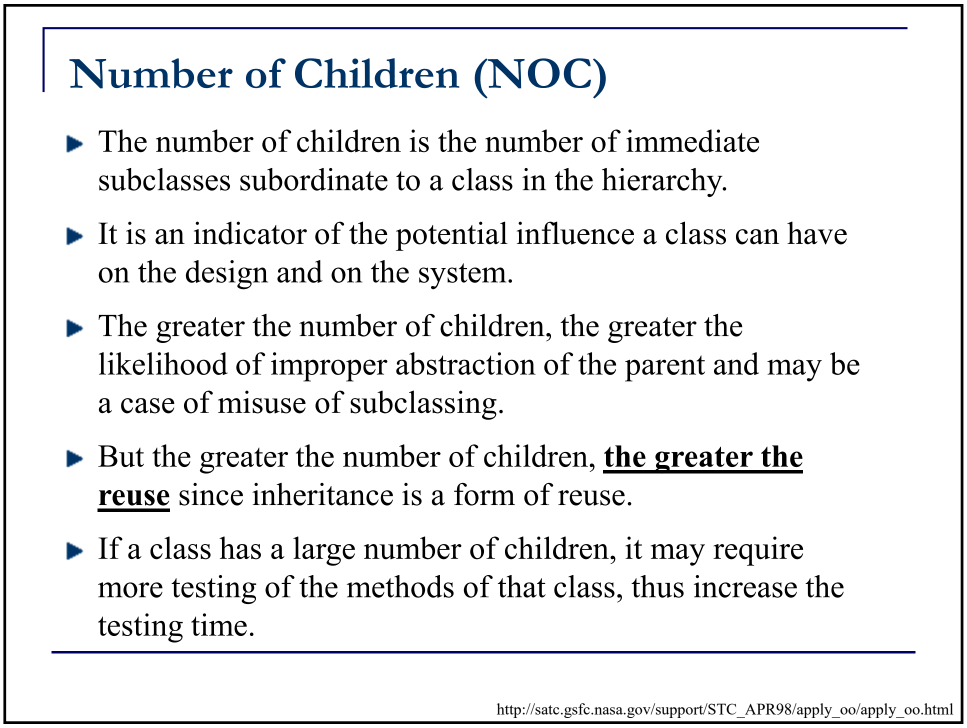
- The depth of a class within the inheritance hierarchy is the maximum number of steps from the class node to the root of the tree and is measured by the number of ancestor classes.
- The deeper a class is within the hierarchy, the greater the number methods it is likely to inherit making it **more complex to predict its behavior.**
- **Deeper trees constitute greater design complexity**, since more methods and classes are involved, but the greater the potential for reuse of inherited methods.
- A support metric for DIT is the number of methods inherited (NMI)

http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



Depth of Inheritance Tree (DIT)

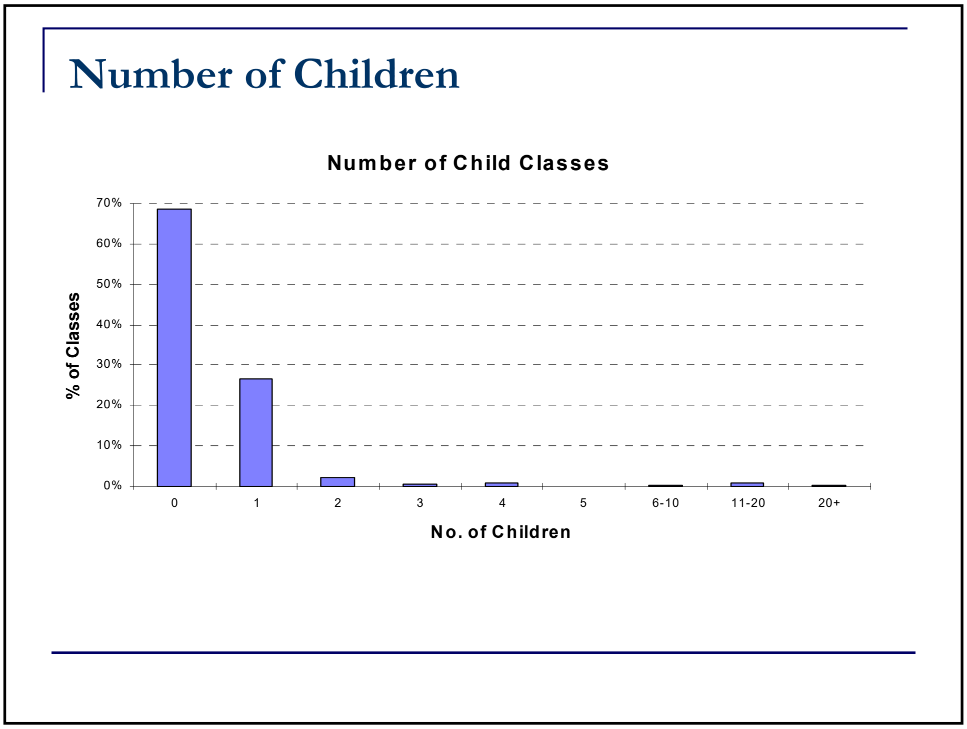
http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



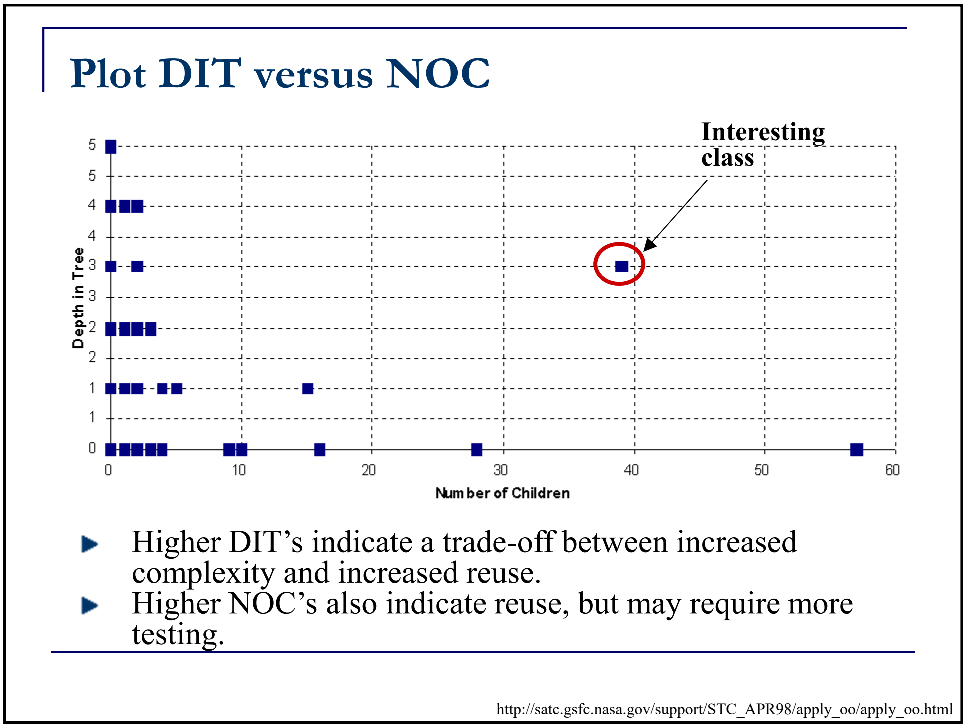
Number of Children (NOC)

- The number of children is the number of immediate subclasses subordinate to a class in the hierarchy.
- It is an indicator of the potential influence a class can have on the design and on the system.
- The greater the number of children, the greater the likelihood of improper abstraction of the parent and may be a case of misuse of subclassing.
- But the greater the number of children, **the greater the reuse** since inheritance is a form of reuse.
- If a class has a large number of children, it may require more testing of the methods of that class, thus increase the testing time.

http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html



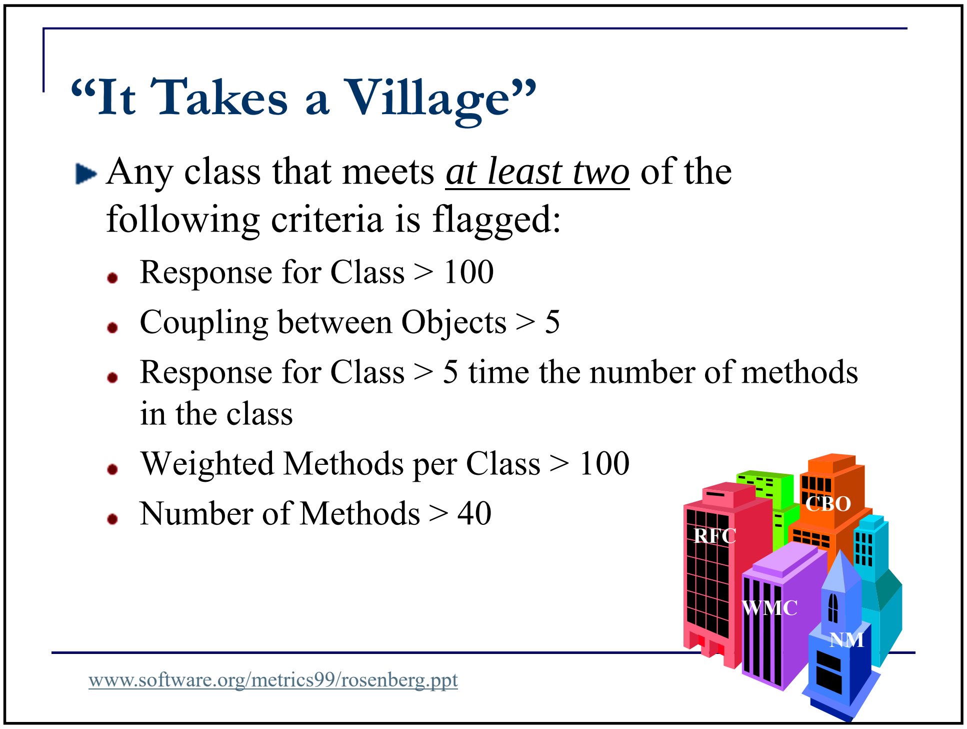
Number of Children



Plot DIT versus NOC

- Higher DIT's indicate a trade-off between increased complexity and increased reuse.
- Higher NOC's also indicate reuse, but may require more testing.

http://satc.gsfc.nasa.gov/support/STC_APR98/apply_oo/apply_oo.html

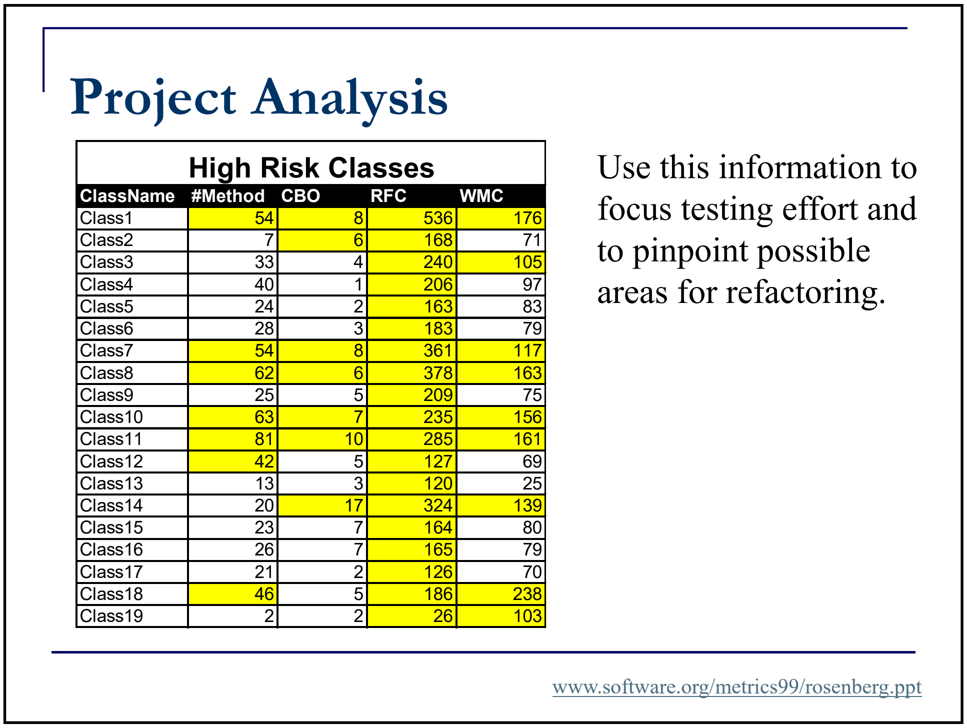


“It Takes a Village”

- Any class that meets *at least two* of the following criteria is flagged:
  - Response for Class > 100
  - Coupling between Objects > 5
  - Response for Class > 5 time the number of methods in the class
  - Weighted Methods per Class > 100
  - Number of Methods > 40

RFC
CBO
WMC
NM

www.software.org/metrics99/rosenberg.ppt



Project Analysis

High Risk Classes

ClassName	#Method	CBO	RFC	WMC
Class1	54	8	536	176
Class2	7	6	168	71
Class3	33	4	240	105
Class4	40	1	206	97
Class5	24	2	163	83
Class6	28	3	183	79
Class7	54	8	361	117
Class8	62	6	378	163
Class9	25	5	209	75
Class10	63	7	235	156
Class11	81	10	285	161
Class12	42	5	127	69
Class13	13	3	120	25
Class14	20	17	324	139
Class15	23	7	164	80
Class16	26	7	165	79
Class17	21	2	126	70
Class18	46	5	186	238
Class19	2	2	26	103

Use this information to focus testing effort and to pinpoint possible areas for refactoring.

www.software.org/metrics99/rosenberg.ppt