Software Size Estimation I

Material adapted from:
Disciplined Software Engineering
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Size Estimating

Why estimate size?

Some estimating background

Size estimating principles

Estimating approaches

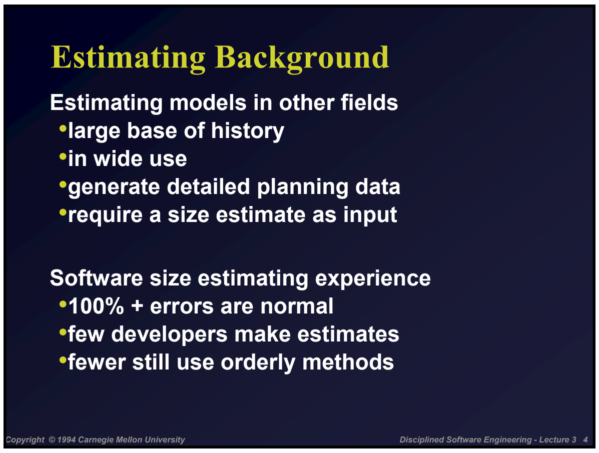
Estimating proxies

Why Estimate Size?

To make better plans
- to better size the job
- to divide the job into separable elements

To assist in tracking progress
- can judge when job scope changes
- can better measure the work

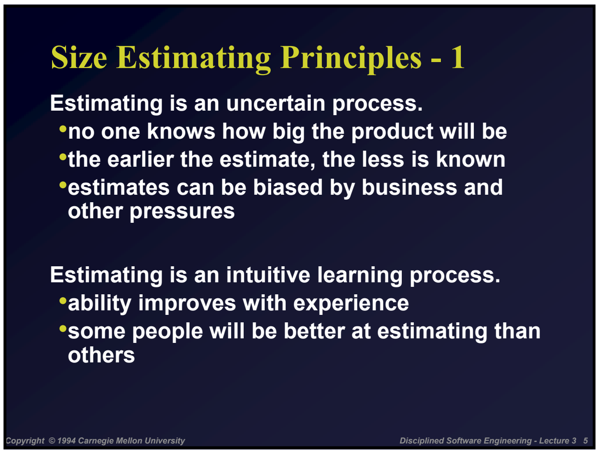
Value in this course
- learn estimating methods
- build estimating skills



Estimating Background

Estimating models in other fields
- large base of history
- in wide use
- generate detailed planning data
- require a size estimate as input

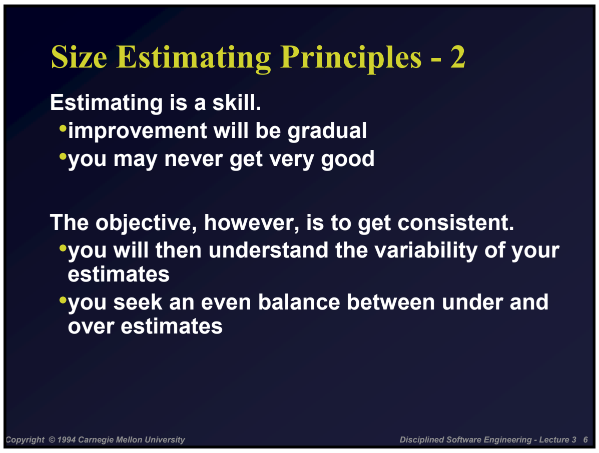
Software size estimating experience
- 100% + errors are normal
- few developers make estimates
- fewer still use orderly methods



Size Estimating Principles - 1

Estimating is an uncertain process.
- no one knows how big the product will be
- the earlier the estimate, the less is known
- estimates can be biased by business and other pressures

Estimating is an intuitive learning process.
- ability improves with experience
- some people will be better at estimating than others

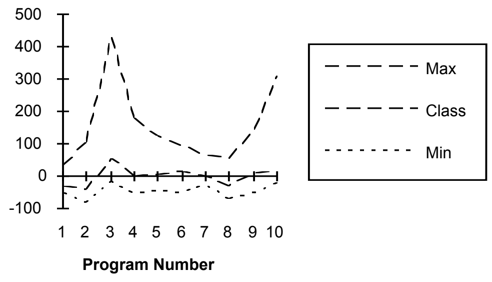


Size Estimating Principles - 2

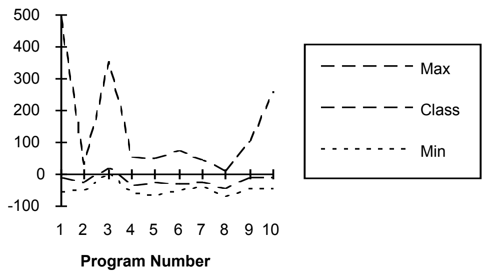
Estimating is a skill.
- improvement will be gradual
- you may never get very good

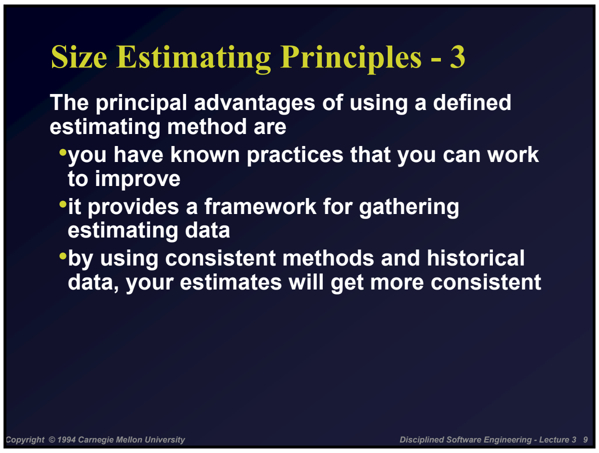
The objective, however, is to get consistent.
- you will then understand the variability of your estimates
- you seek an even balance between under and over estimates

Size Estimating Errors - 12 Students



Time Estimating Errors - 12 Students

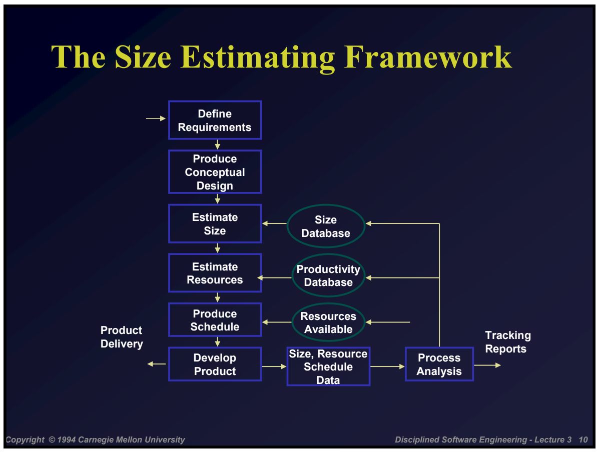




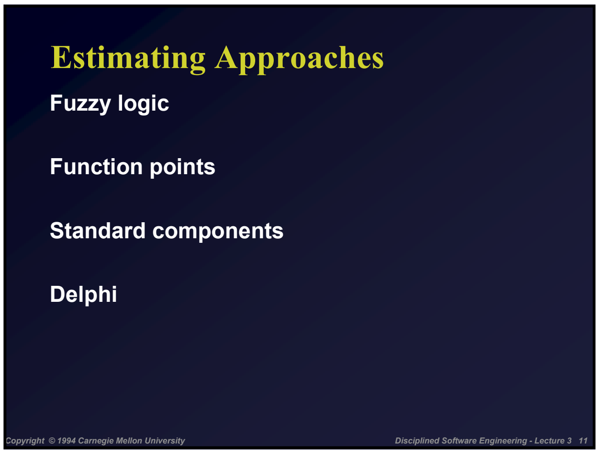
Size Estimating Principles - 3

The principal advantages of using a defined estimating method are

- you have known practices that you can work to improve
- it provides a framework for gathering estimating data
- by using consistent methods and historical data, your estimates will get more consistent



The Size Estimating Framework



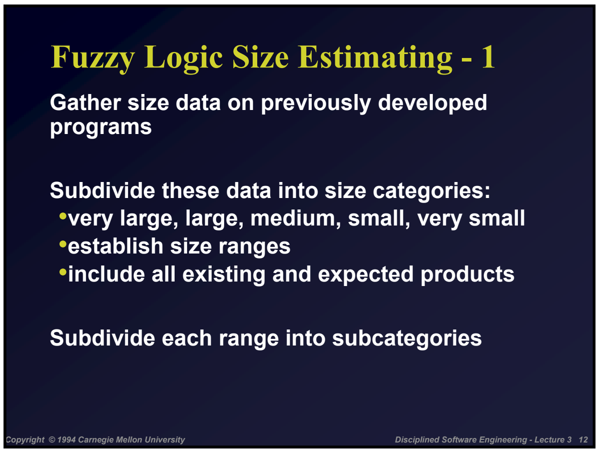
Estimating Approaches

Fuzzy logic

Function points

Standard components

Delphi

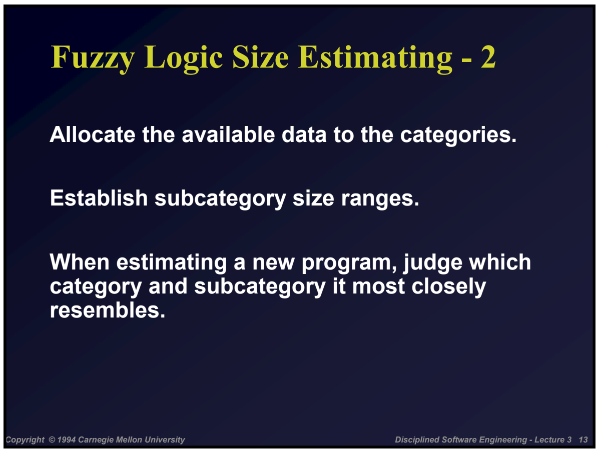


Fuzzy Logic Size Estimating - 1

Gather size data on previously developed programs

Subdivide these data into size categories:
- very large, large, medium, small, very small
- establish size ranges
- include all existing and expected products

Subdivide each range into subcategories

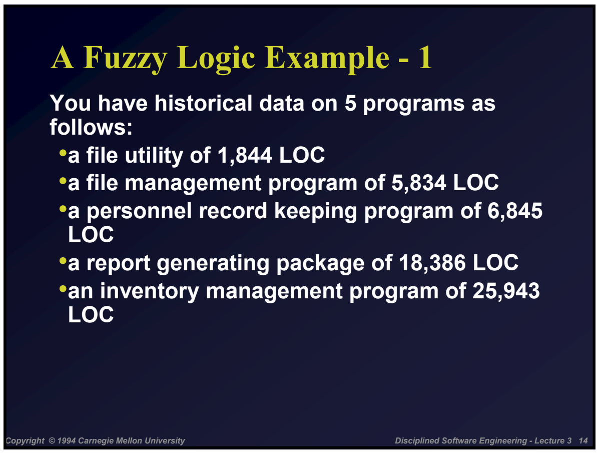


Fuzzy Logic Size Estimating - 2

Allocate the available data to the categories.

Establish subcategory size ranges.

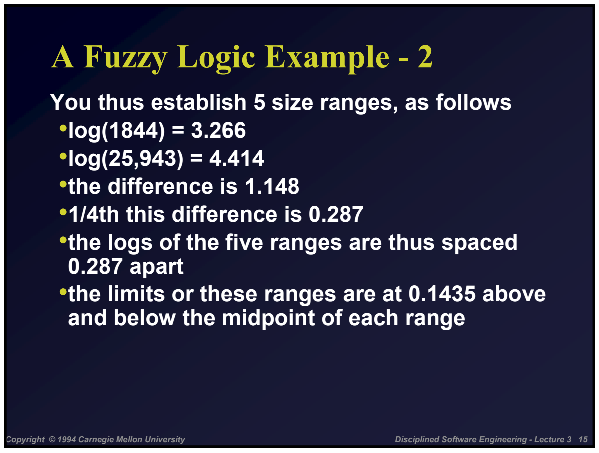
When estimating a new program, judge which category and subcategory it most closely resembles.



A Fuzzy Logic Example - 1

You have historical data on 5 programs as follows:

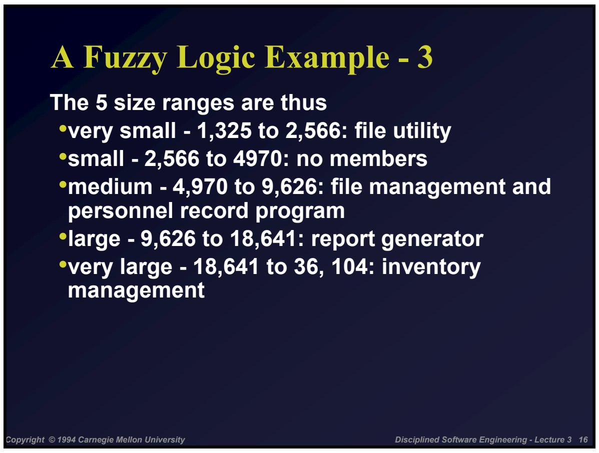
- a file utility of 1,844 LOC
- a file management program of 5,834 LOC
- a personnel record keeping program of 6,845 LOC
- a report generating package of 18,386 LOC
- an inventory management program of 25,943 LOC



A Fuzzy Logic Example - 2

You thus establish 5 size ranges, as follows

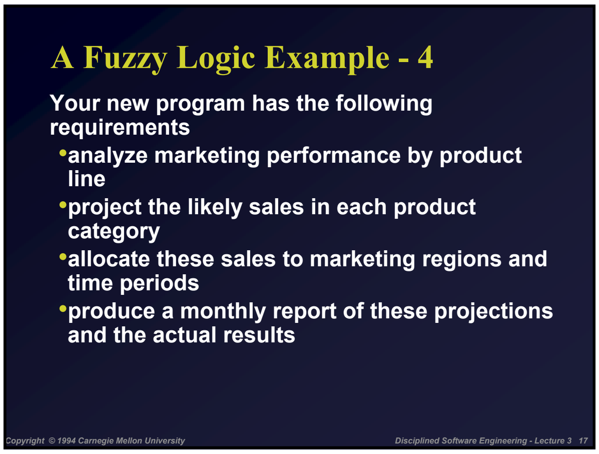
- log(1844) = 3.266
- log(25,943) = 4.414
- the difference is 1.148
- 1/4th this difference is 0.287
- the logs of the five ranges are thus spaced 0.287 apart
- the limits or these ranges are at 0.1435 above and below the midpoint of each range



A Fuzzy Logic Example - 3

The 5 size ranges are thus

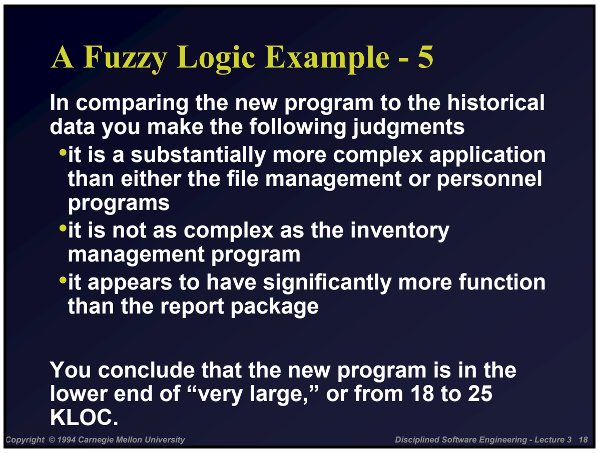
- very small - 1,325 to 2,566: file utility
- small - 2,566 to 4970: no members
- medium - 4,970 to 9,626: file management and personnel record program
- large - 9,626 to 18,641: report generator
- very large - 18,641 to 36, 104: inventory management



A Fuzzy Logic Example - 4

Your new program has the following requirements

- analyze marketing performance by product line
- project the likely sales in each product category
- allocate these sales to marketing regions and time periods
- produce a monthly report of these projections and the actual results

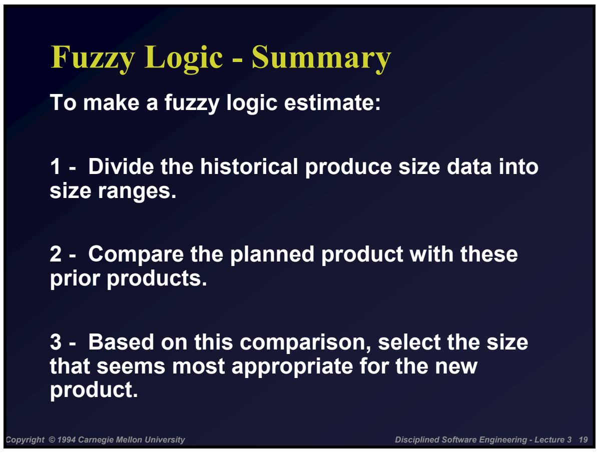


A Fuzzy Logic Example - 5

In comparing the new program to the historical data you make the following judgments

- it is a substantially more complex application than either the file management or personnel programs
- it is not as complex as the inventory management program
- it appears to have significantly more function than the report package

You conclude that the new program is in the lower end of "very large," or from 18 to 25 KLOC.



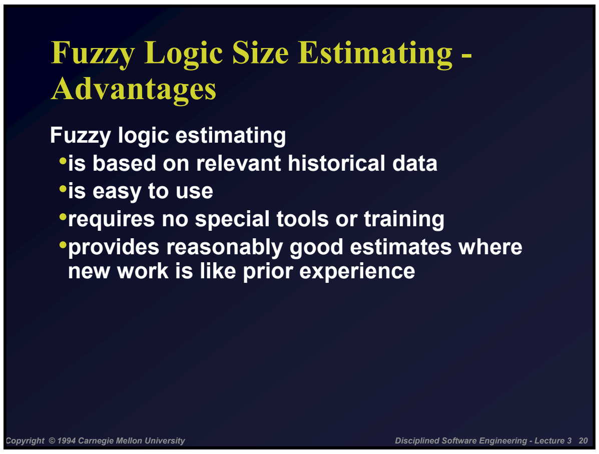
Fuzzy Logic - Summary

To make a fuzzy logic estimate:

1 - Divide the historical produce size data into size ranges.

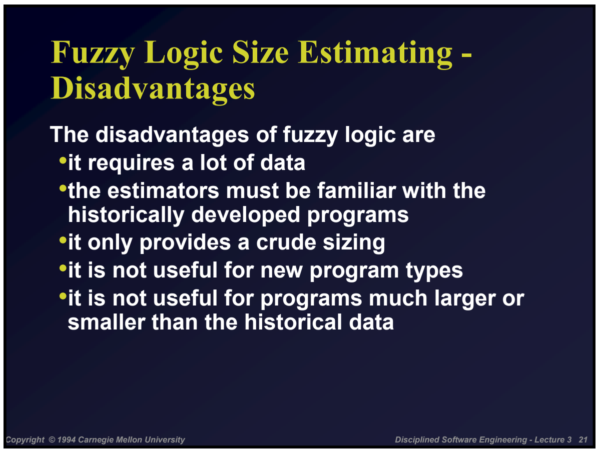
2 - Compare the planned product with these prior products.

3 - Based on this comparison, select the size that seems most appropriate for the new product.



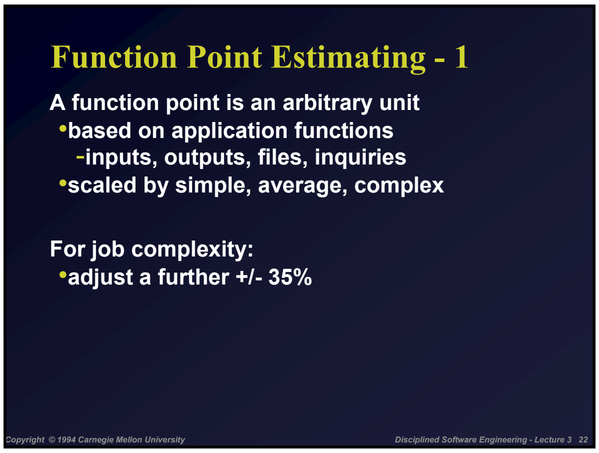
Fuzzy Logic Size Estimating - Advantages

Fuzzy logic estimating
- is based on relevant historical data
- is easy to use
- requires no special tools or training
- provides reasonably good estimates where new work is like prior experience



Fuzzy Logic Size Estimating - Disadvantages

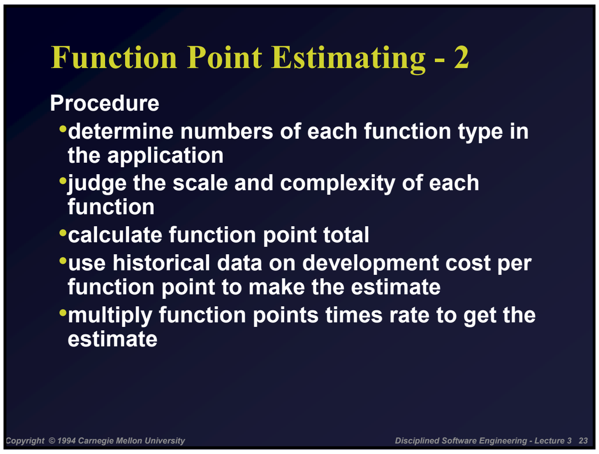
The disadvantages of fuzzy logic are
- it requires a lot of data
- the estimators must be familiar with the historically developed programs
- it only provides a crude sizing
- it is not useful for new program types
- it is not useful for programs much larger or smaller than the historical data



Function Point Estimating - 1

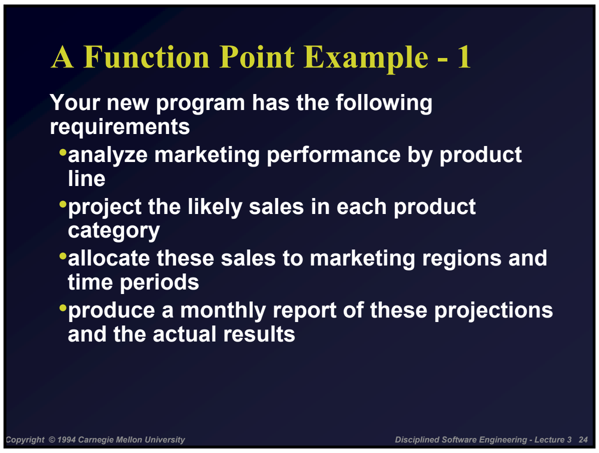
A function point is an arbitrary unit
- based on application functions
  - inputs, outputs, files, inquiries
- scaled by simple, average, complex

For job complexity:
- adjust a further +/- 35%



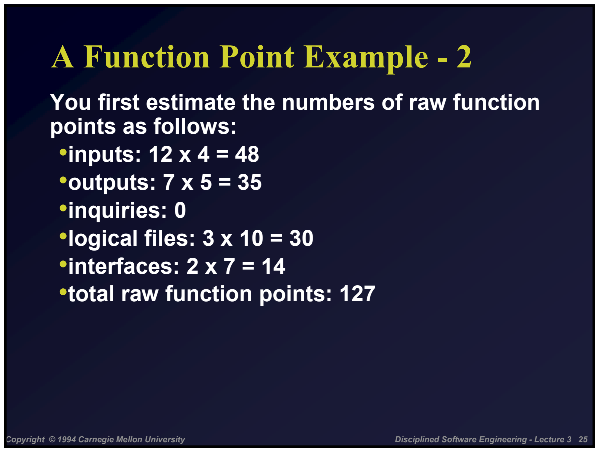
Function Point Estimating - 2

Procedure
- determine numbers of each function type in the application
- judge the scale and complexity of each function
- calculate function point total
- use historical data on development cost per function point to make the estimate
- multiply function points times rate to get the estimate



A Function Point Example - 1

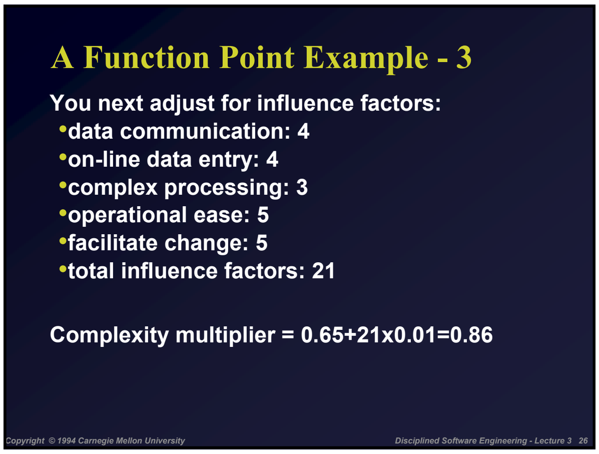
Your new program has the following requirements
- analyze marketing performance by product line
- project the likely sales in each product category
- allocate these sales to marketing regions and time periods
- produce a monthly report of these projections and the actual results



A Function Point Example - 2

You first estimate the numbers of raw function points as follows:

- inputs: 12 x 4 = 48
- outputs: 7 x 5 = 35
- inquiries: 0
- logical files: 3 x 10 = 30
- interfaces: 2 x 7 = 14
- total raw function points: 127

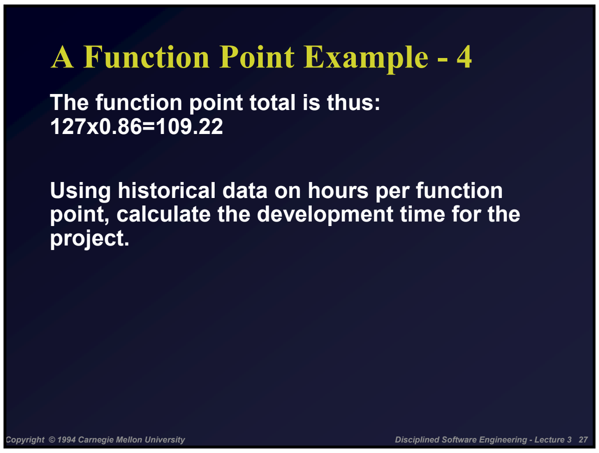


A Function Point Example - 3

You next adjust for influence factors:

- data communication: 4
- on-line data entry: 4
- complex processing: 3
- operational ease: 5
- facilitate change: 5
- total influence factors: 21

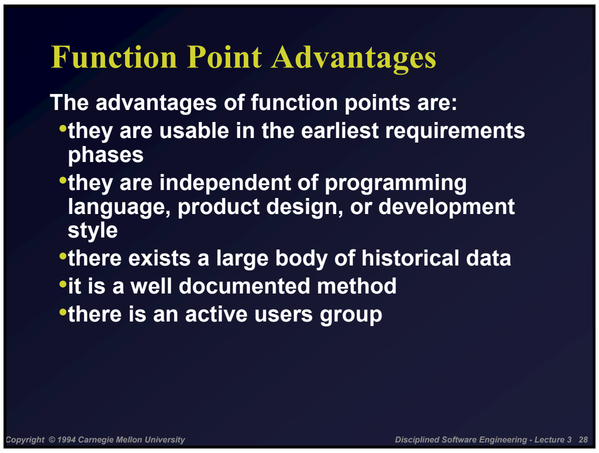
Complexity multiplier = 0.65+21x0.01=0.86



A Function Point Example - 4

The function point total is thus: 127x0.86=109.22

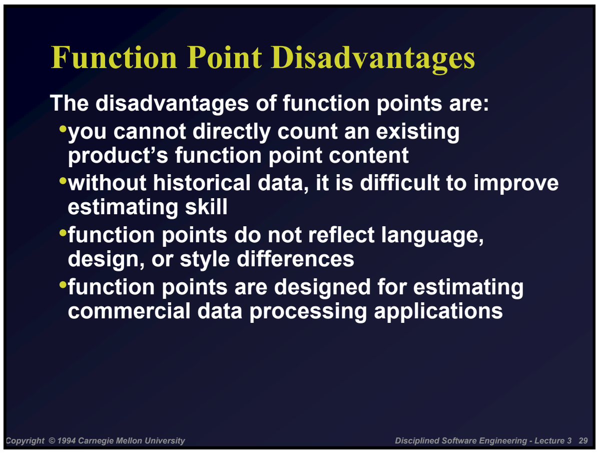
Using historical data on hours per function point, calculate the development time for the project.



Function Point Advantages

The advantages of function points are:

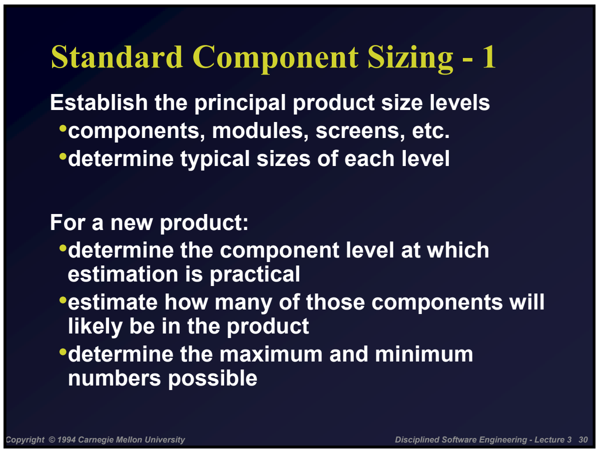
- they are usable in the earliest requirements phases
- they are independent of programming language, product design, or development style
- there exists a large body of historical data
- it is a well documented method
- there is an active users group



Function Point Disadvantages

The disadvantages of function points are:

- you cannot directly count an existing product's function point content
- without historical data, it is difficult to improve estimating skill
- function points do not reflect language, design, or style differences
- function points are designed for estimating commercial data processing applications



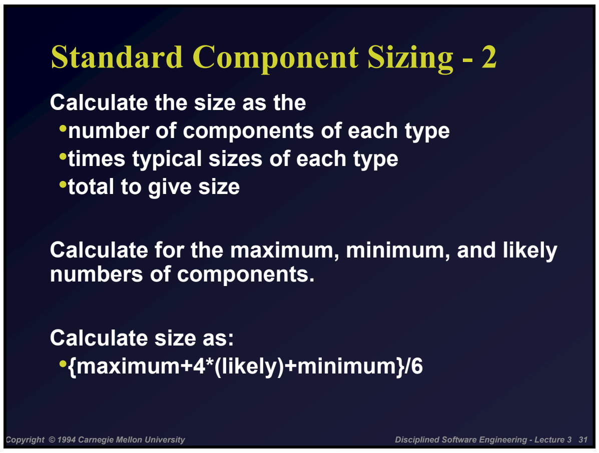
Standard Component Sizing - 1

Establish the principal product size levels

- components, modules, screens, etc.
- determine typical sizes of each level

For a new product:

- determine the component level at which estimation is practical
- estimate how many of those components will likely be in the product
- determine the maximum and minimum numbers possible

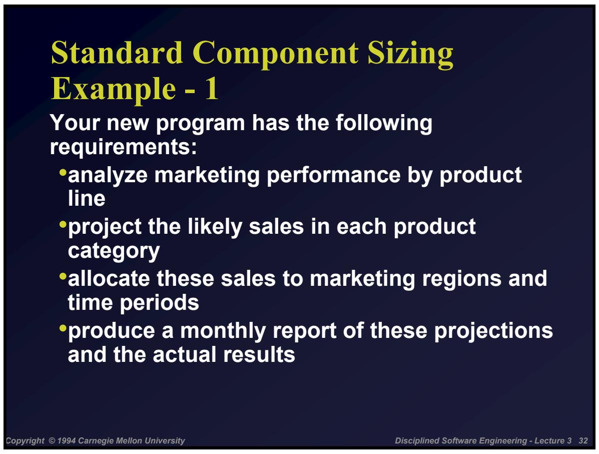


Standard Component Sizing - 2

Calculate the size as the
- number of components of each type
- times typical sizes of each type
- total to give size

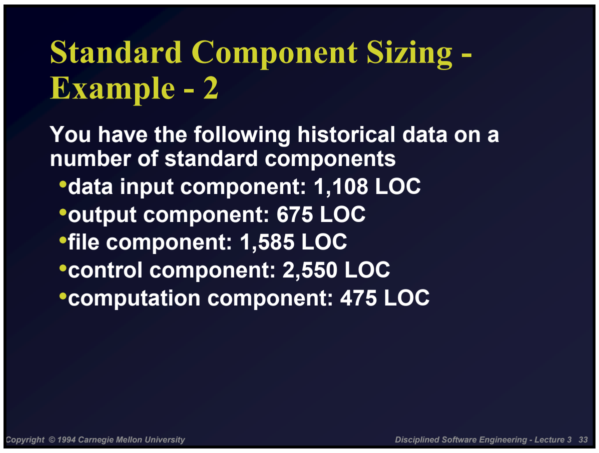
Calculate for the maximum, minimum, and likely numbers of components.

Calculate size as:
- {maximum+4*(likely)+minimum}/6



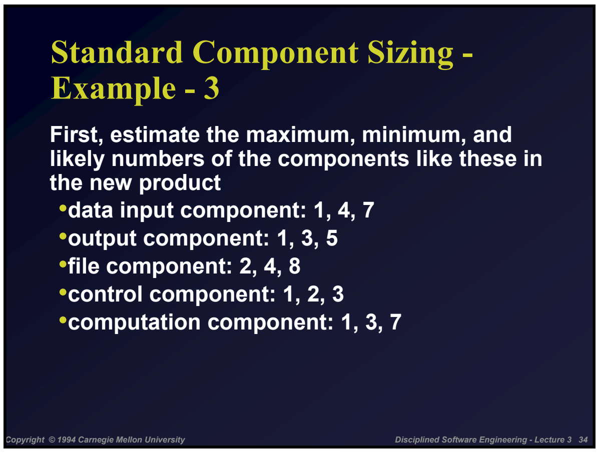
Standard Component Sizing Example - 1

Your new program has the following requirements:
- analyze marketing performance by product line
- project the likely sales in each product category
- allocate these sales to marketing regions and time periods
- produce a monthly report of these projections and the actual results



Standard Component Sizing - Example - 2

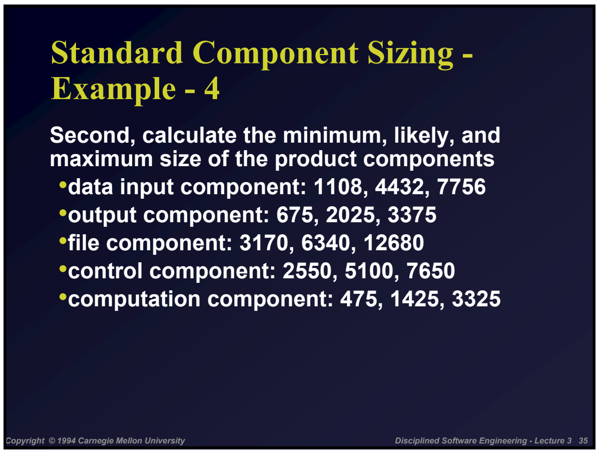
You have the following historical data on a number of standard components
- data input component: 1,108 LOC
- output component: 675 LOC
- file component: 1,585 LOC
- control component: 2,550 LOC
- computation component: 475 LOC



Standard Component Sizing - Example - 3

First, estimate the maximum, minimum, and likely numbers of the components like these in the new product

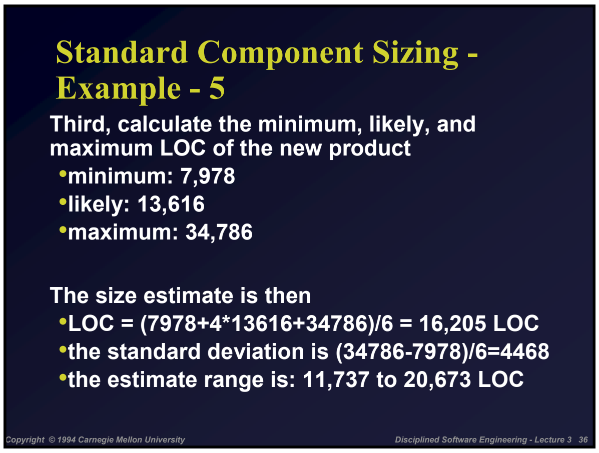
- data input component: 1, 4, 7
- output component: 1, 3, 5
- file component: 2, 4, 8
- control component: 1, 2, 3
- computation component: 1, 3, 7



Standard Component Sizing - Example - 4

Second, calculate the minimum, likely, and maximum size of the product components

- data input component: 1108, 4432, 7756
- output component: 675, 2025, 3375
- file component: 3170, 6340, 12680
- control component: 2550, 5100, 7650
- computation component: 475, 1425, 3325



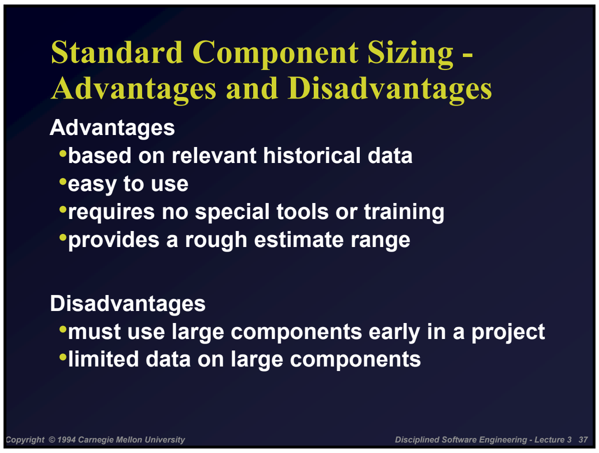
Standard Component Sizing - Example - 5

Third, calculate the minimum, likely, and maximum LOC of the new product

- minimum: 7,978
- likely: 13,616
- maximum: 34,786

The size estimate is then

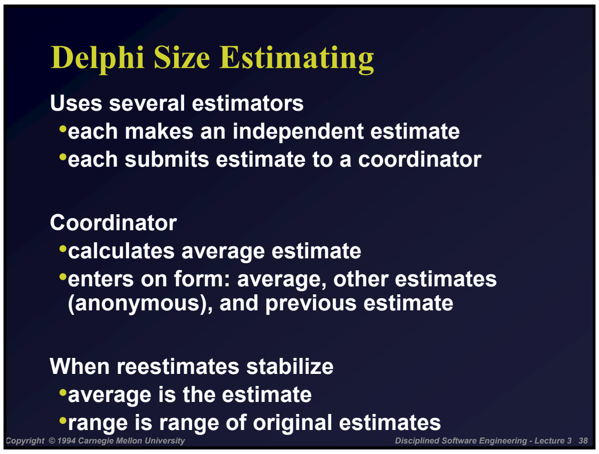
- LOC = (7978+4*13616+34786)/6 = 16,205 LOC
- the standard deviation is (34786-7978)/6=4468
- the estimate range is: 11,737 to 20,673 LOC



Standard Component Sizing - Advantages and Disadvantages

Advantages
- based on relevant historical data
- easy to use
- requires no special tools or training
- provides a rough estimate range

Disadvantages
- must use large components early in a project
- limited data on large components

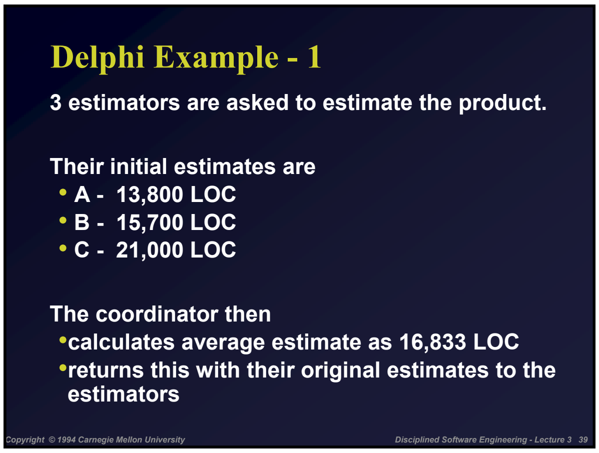


Delphi Size Estimating

Uses several estimators
- each makes an independent estimate
- each submits estimate to a coordinator

Coordinator
- calculates average estimate
- enters on form: average, other estimates (anonymous), and previous estimate

When reestimates stabilize
- average is the estimate
- range is range of original estimates

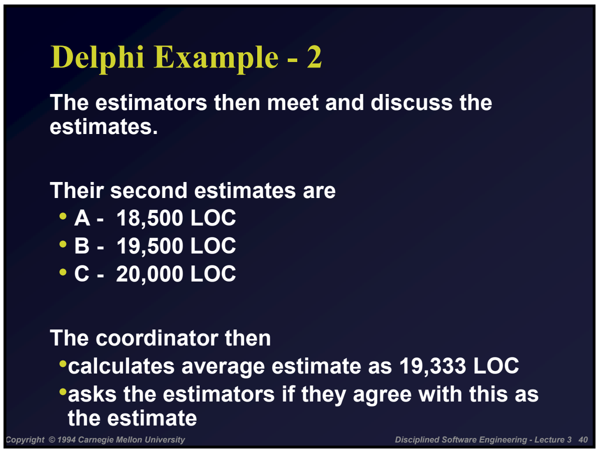


Delphi Example - 1

3 estimators are asked to estimate the product.

Their initial estimates are
- A - 13,800 LOC
- B - 15,700 LOC
- C - 21,000 LOC

The coordinator then
- calculates average estimate as 16,833 LOC
- returns this with their original estimates to the estimators

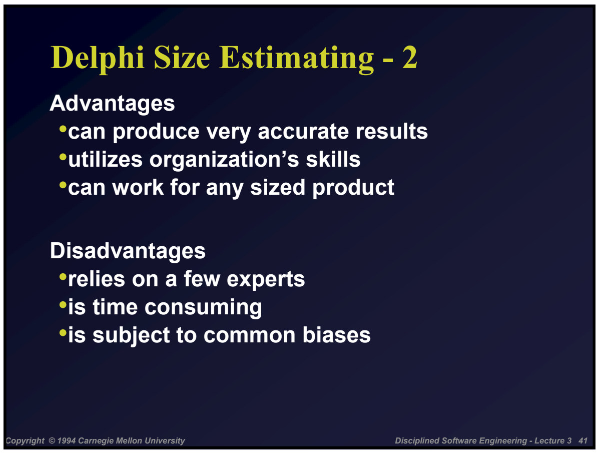


Delphi Example - 2

The estimators then meet and discuss the estimates.

Their second estimates are
- A - 18,500 LOC
- B - 19,500 LOC
- C - 20,000 LOC

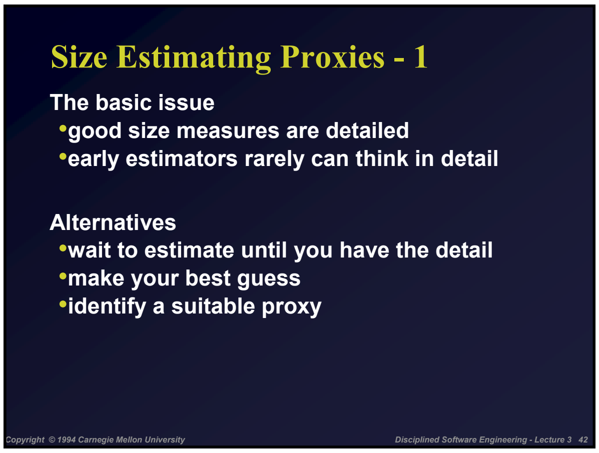
The coordinator then
- calculates average estimate as 19,333 LOC
- asks the estimators if they agree with this as the estimate



Delphi Size Estimating - 2

Advantages
- can produce very accurate results
- utilizes organization's skills
- can work for any sized product

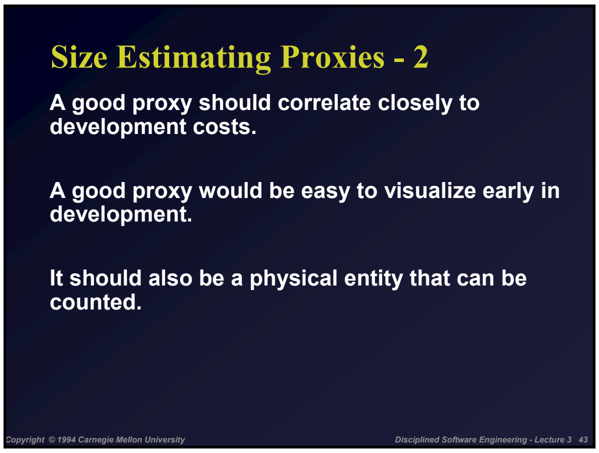
Disadvantages
- relies on a few experts
- is time consuming
- is subject to common biases



Size Estimating Proxies - 1

The basic issue
- good size measures are detailed
- early estimators rarely can think in detail

Alternatives
- wait to estimate until you have the detail
- make your best guess
- identify a suitable proxy

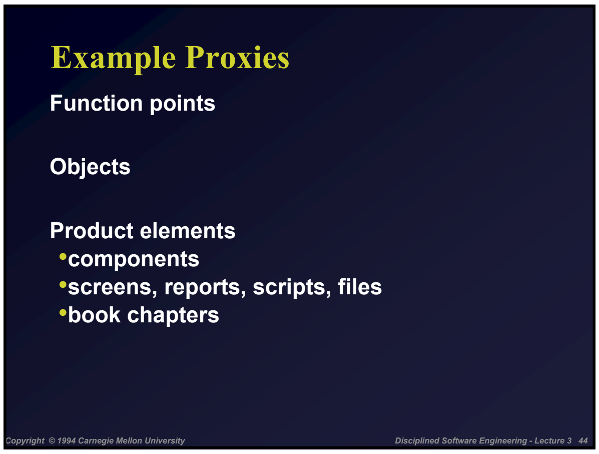


Size Estimating Proxies - 2

A good proxy should correlate closely to development costs.

A good proxy would be easy to visualize early in development.

It should also be a physical entity that can be counted.



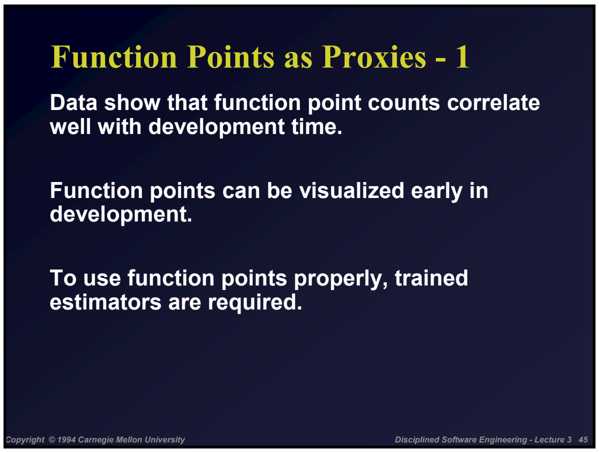
Example Proxies

Function points

Objects

Product elements

- components
- screens, reports, scripts, files
- book chapters

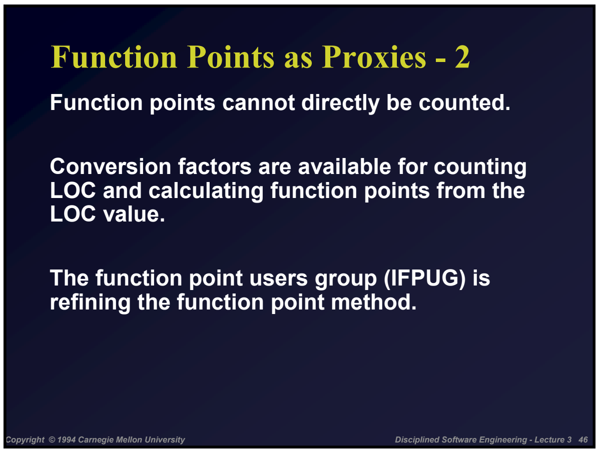


Function Points as Proxies - 1

Data show that function point counts correlate well with development time.

Function points can be visualized early in development.

To use function points properly, trained estimators are required.

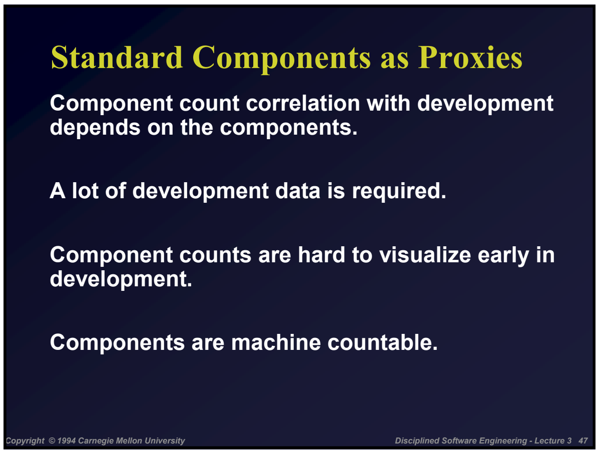


Function Points as Proxies - 2

Function points cannot directly be counted.

Conversion factors are available for counting LOC and calculating function points from the LOC value.

The function point users group (IFPUG) is refining the function point method.



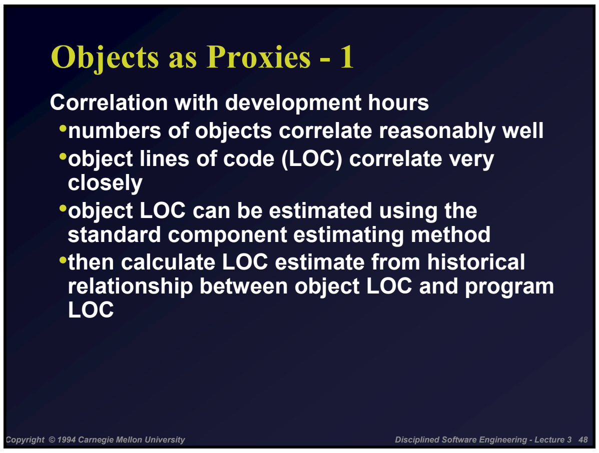
Standard Components as Proxies

Component count correlation with development depends on the components.

A lot of development data is required.

Component counts are hard to visualize early in development.

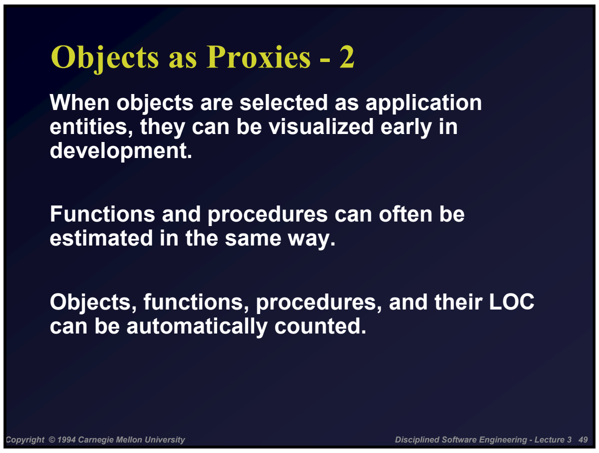
Components are machine countable.



Objects as Proxies - 1

Correlation with development hours

- numbers of objects correlate reasonably well
- object lines of code (LOC) correlate very closely
- object LOC can be estimated using the standard component estimating method
- then calculate LOC estimate from historical relationship between object LOC and program LOC



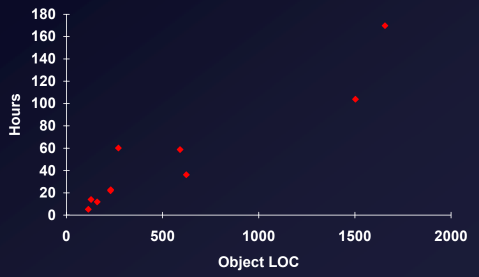
Objects as Proxies - 2

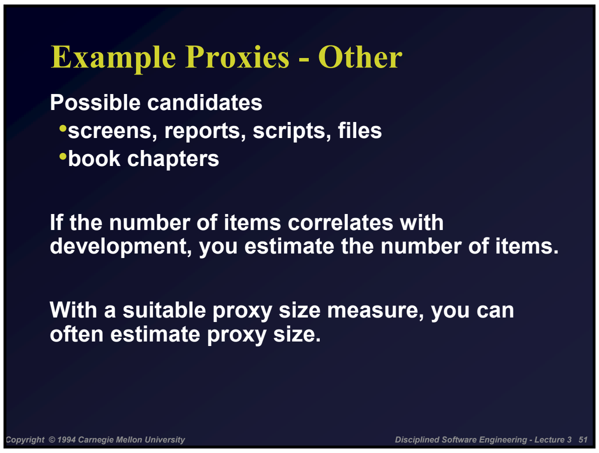
When objects are selected as application entities, they can be visualized early in development.

Functions and procedures can often be estimated in the same way.

Objects, functions, procedures, and their LOC can be automatically counted.

Object LOC Correlation With Development Hours



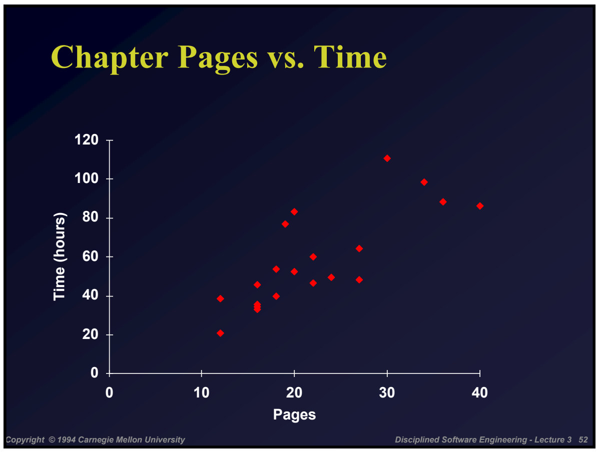


Example Proxies - Other

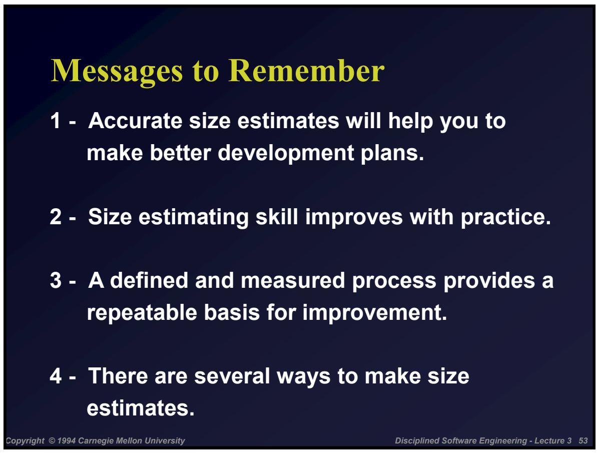
Possible candidates
- screens, reports, scripts, files
- book chapters

If the number of items correlates with development, you estimate the number of items.

With a suitable proxy size measure, you can often estimate proxy size.



Chapter Pages vs. Time



Messages to Remember

1 - Accurate size estimates will help you to make better development plans.

2 - Size estimating skill improves with practice.

3 - A defined and measured process provides a repeatable basis for improvement.

4 - There are several ways to make size estimates.