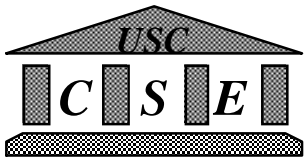


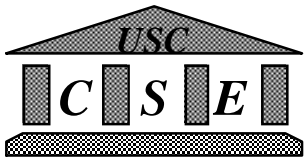
# **COCOMO II Overview**

**CS 510, 577a**

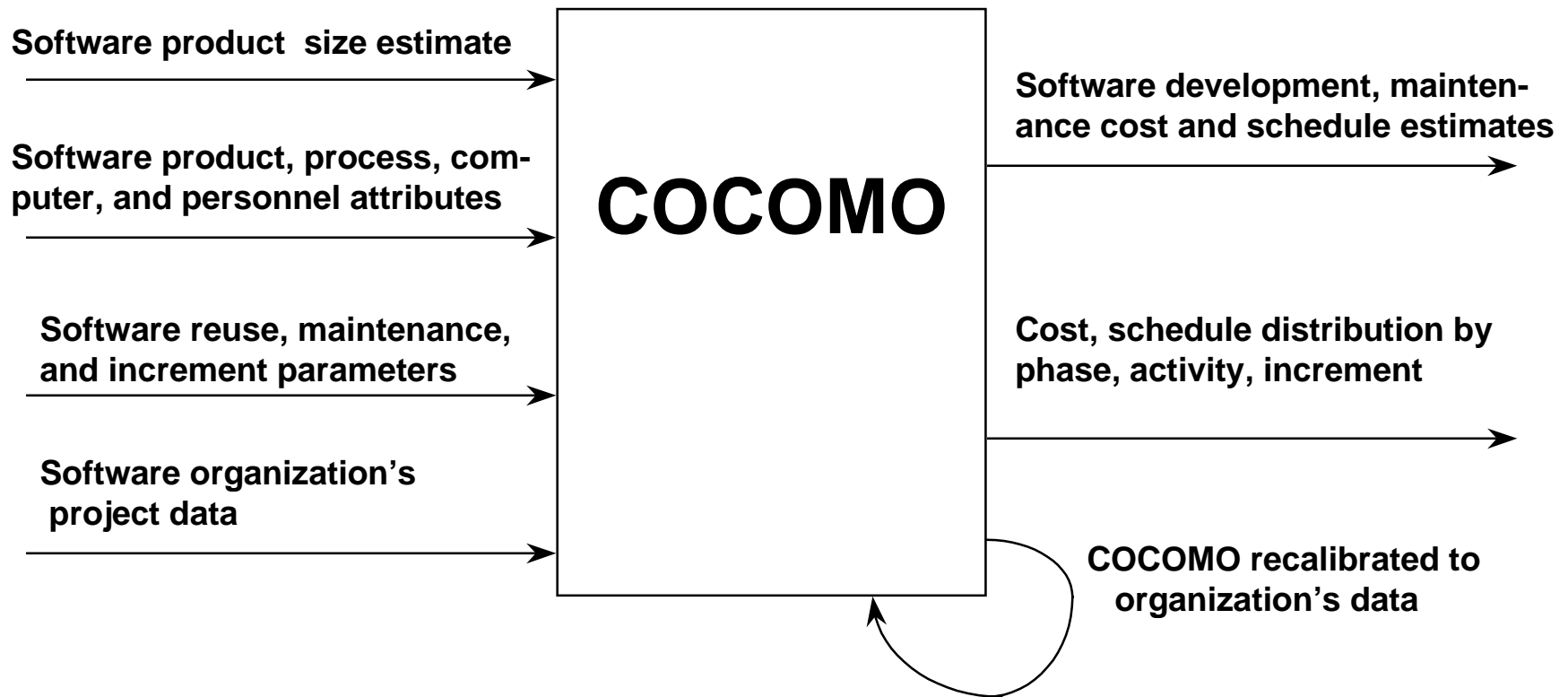


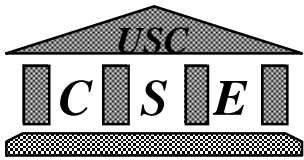
# COCOMO II Model Overview

- **COCOMO Baseline Overview**
- **COCOMO II Objectives**
- **Coverage of Future Market Sectors**
- **Hierarchical Sizing Model**
- **Modes Replaced by Exponent Drivers**
- **Stronger Reuse/Reengineering Model**
- **Other Model Improvements**



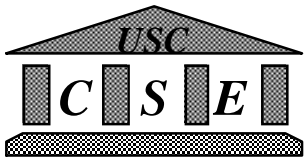
# COCOMO Baseline Overview I





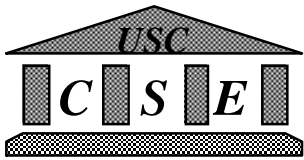
# COCOMO Baseline Overview II

- **Open interfaces and internals**
  - **Published in “Software Engineering Economics,”  
Boehm, 1981**
- **Numerous implementations, calibrations, extensions**
  - **Incremental development, Ada, new environment  
technology**
  - **Arguably the most frequently-used software cost  
model worldwide.**
- **Needs reengineering to address new estimation challenges**

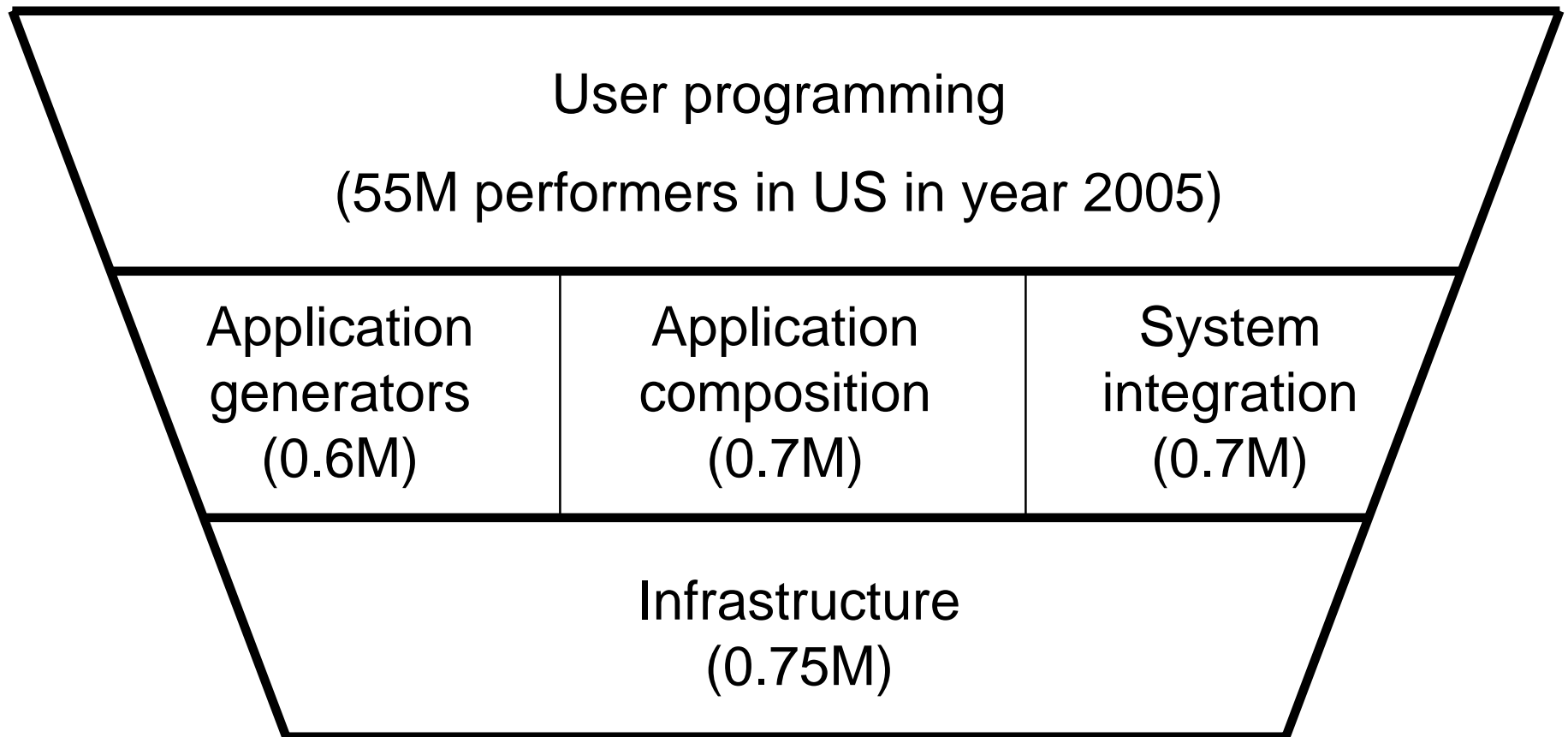


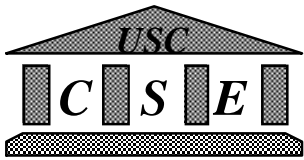
# COCOMO II Objectives

- **Develop a 1990's-2000's software cost model**
  - **Addressing new processes and practices**
- **Retain COCOMO internal, external openness**
- **Develop database, tool support for continuous model improvement**
- **Support closed-loop quantitative project management and process improvement**



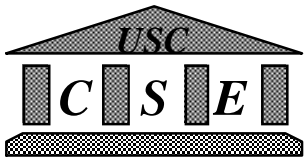
# The future of the software practices marketplace





# COCOMO II Coverage of Future SW Practices Sectors

- **User Programming: No need for cost model**
- **Applications Composition: Use object counts or object points**
  - **Count (weight) screens, reports, 3GL routines**
- **System Integration; development of applications generators and infrastructure software**
  - **Prototyping: Applications composition model**
  - **Early design: Function Points and/or Source Statements and 7 cost drivers**
  - **Post-architecture: Source Statements and/or Function Points and 17 cost drivers**
  - **Stronger reuse/reengineering model**



# Baseline Object Point Estimation Procedure

- Step 1: Assess Object-Counts: estimate the number of screens, reports, and 3GL components that will comprise this application. Assume the standard definitions of these objects in your ICASE environment.
- Step 2: Classify each object instance into simple, medium and difficult complexity levels depending on values of characteristic dimensions. Use the following scheme:

For Screens				For Reports			
# and source of data tables				# and source of data tables			
Number of Views Contained	Total < 4 (<2 svr, <3 clnt)	Total <8 (<3 svr, 3 - 5 clnt)	Total 8+ (>3 svr, >5 clnt)	Number of Sections Contained	Total < 4 (<2 svr, <3 clnt)	Total <8 (<3 svr, 3 - 5 clnt)	Total 8+ (>3 svr, >5 clnt)
<3	simple	simple	medium	0 or 1	simple	simple	medium
3-7	simple	medium	difficult	2 or 3	simple	medium	difficult
>8	medium	difficult	difficult	4+	medium	difficult	difficult

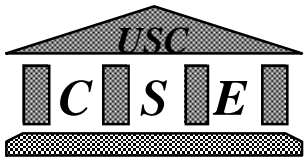
- Step 3: Weigh the number in each cell using the following scheme. The weights reflect the relative effort required to implement an instance of that complexity level.

Object Type	Complexity-Weight		
	Simple	Medium	Difficult
Screen	1	2	3
Report	2	5	8
3GL Component			10

- Step 4: Determine Object-Points: add all the weighted object instances to get one number, the Object-Point count.
- Step 5: Estimate percentage of reuse you expect to be achieved in this project. Compute the New Object Points to be developed  $NOP = (Object-Points) (100 - \%reuse) / 100$ .
- Step 6: Determine a productivity rate,  $PROD = NOP / person-month$ , from the following scheme:

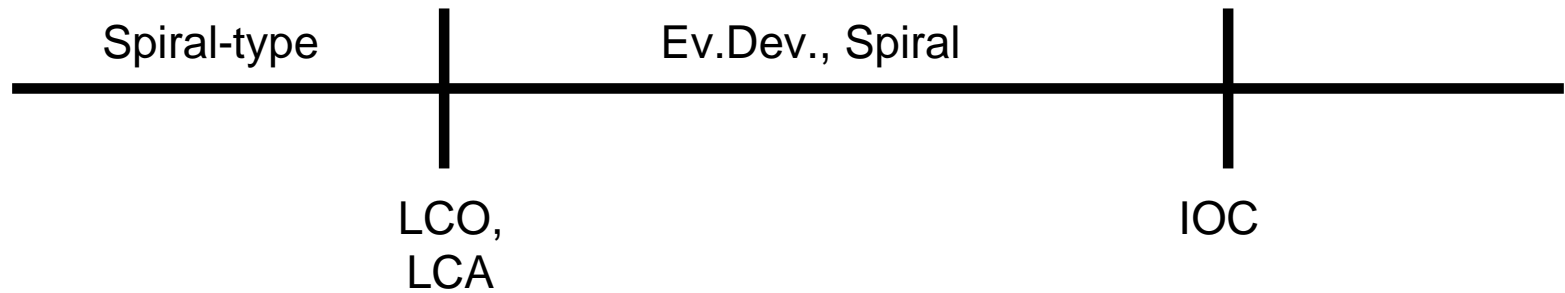
Developer's experience and capability	Very Low	Low	Nominal	High	Very High
ICASE maturity and capability	Very Low	Low	Nominal	High	Very High
PROD	4	7	13	25	50

- Step 7: Compute the estimated person-months:  $PM = NOP / PROD$ .

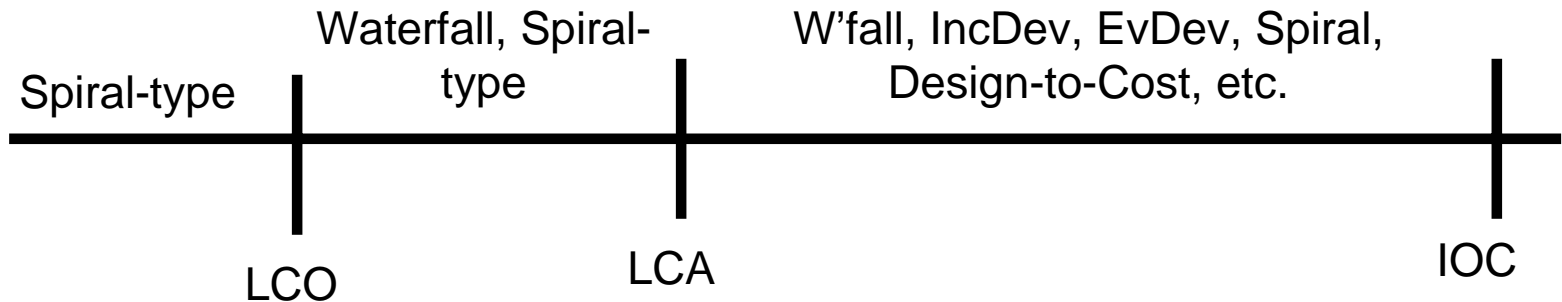


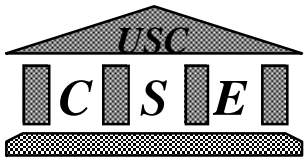
# Process/Anchor Point Examples

Rapid  
App.  
Devel.



Sys  
Devel





# Early Design and Post-Arch Model

- **Effort:**

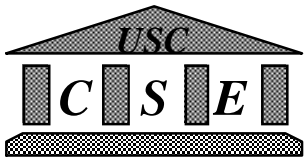
$$PM_{estimated} = A \times (Size)^{(SF)} \times \left\{ \prod_i EM_i \right\}$$

- **Size**

- KSLOC (Thousands of Source Lines of Code)
- UFP (Unadjusted Function Points)
- EKSLOC (Equivalent KSLOC) used for adaptation

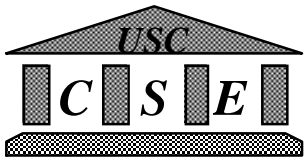
- **SF: Scale Factors (5)**

- **EM: Effort Multipliers (7 for ED, 17 for PA)**



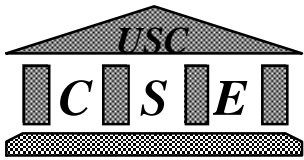
# COCOMO II Model Overview

- COCOMO Baseline Overview
- COCOMO II Objectives
- Coverage of Future Market Sectors
- ➔ ● Hierarchical Sizing Model
- Modes Replaced by Exponent Drivers
- Stronger Reuse/Reengineering Model
- Other Model Improvements



# New Scaling Exponent Approach

- **Nominal person-months =  $A * (\text{size})^{**} B$**
- **$B = 1.01 + 0.01 \triangleright (\text{exponent driver ratings})$** 
  - B ranges from 1.01 to 1.26
  - 5 drivers; ratings from 0 to 5
- **Exponent drivers:**
  - Precedentedness
  - Development flexibility
  - Architecture/ risk resolution
  - Team cohesion
  - Process maturity (derived from SEI CMM)

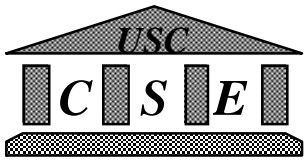


# Project Scale Factors

$$PM_{estimated} = A \times (Size)^{(SF)} \times \left( \prod_i EM_i \right)$$

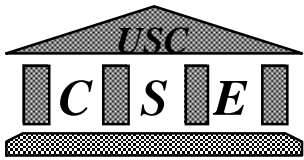
$$SF = 1.01 + 0.01 \times \sum w_i$$

Scale Factors ( <i>W</i> )	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
PMAT	weighted sum of KPA achievement levels					

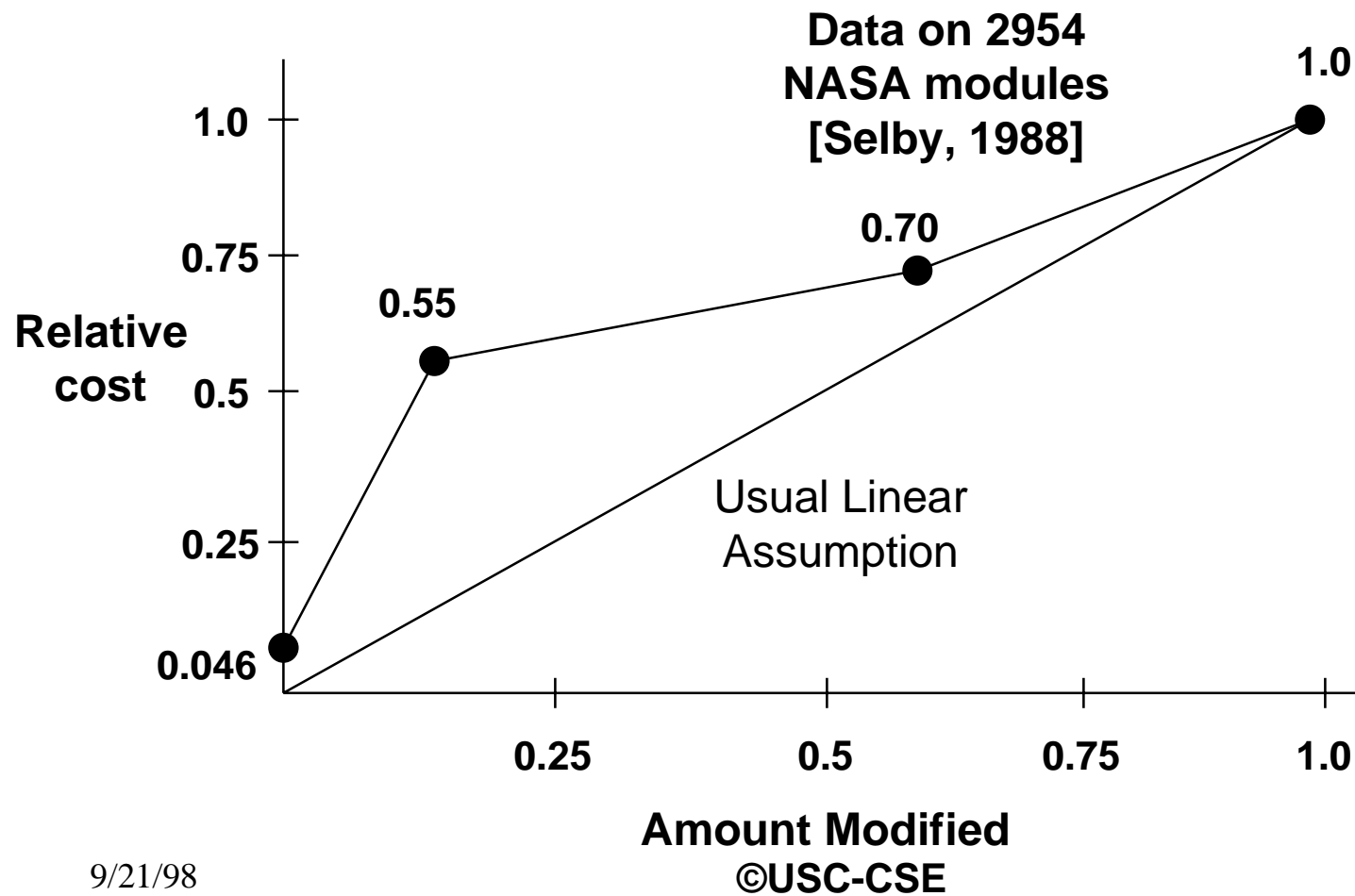


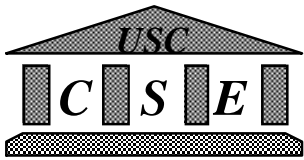
# Reuse and Product Line Management

- **Challenges**
  - Estimate costs of both reusing software and developing software for future reuse
  - Estimate extra effects on schedule (if any)
- **Responses**
  - New nonlinear reuse model
  - Cost of developing reusable software expanded from Ada COCOMO
  - Gathering schedule data



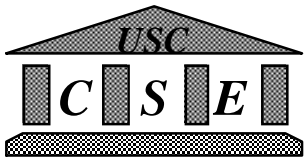
# Nonlinear Reuse Effects





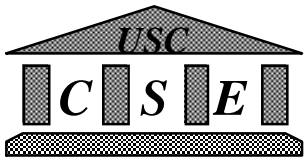
# Reuse and Reengineering Effects

- **Add Assessment & Assimilation increment (AA)**
  - Similar to conversion planning increment
- **Add software understanding increment (SU)**
  - To cover nonlinear software understanding effects
  - Coupled with software unfamiliarity level (UNFM)
  - Apply only if reused software is modified
- **Results in revised Equivalent Source Lines of Code (ESLOC)**
  - $AAF = 0.4(DM) + 0.3(CM) + 0.3(IM)$
  - $ESLOC = ASLOC[AA + AAF(1 + 0.02(SU)(UNFM))]$ ,  
 $AAF \leq 0.5$
  - $ESLOC = ASLOC[AA + AAF(SU)(UNFM)]$ ,  $AAF > 0.5$



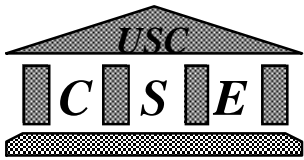
# Software Understanding Rating / Increment

	Very Low	Low	Nom	High	Very High
Structure	Very low cohesion, high coupling, spaghetti code.	Moderately low cohesion, high coupling.	Reasonably well - structured; some weak areas.	High cohesion, low coupling.	Strong modularity, information hiding in data/control structures.
Application Clarity	No match between program and application world views.	Some correlation between program and application .	Moderate correlation between program and application .	Good correlation between program and application .	Clear match between program and application world views.
Self - Descriptiveness	Obscure code; documentation missing, obscure or obsolete.	Some code commentary and headers; some useful documentation.	Moderate level of code commentary, headers, documentation.	Good code commentary and headers; useful documentation; some weak areas.	Self - descriptive code; documentation up-to-date, well-organized, with design rationale.
SU Increment to ESLOC	50	40	30	20	10



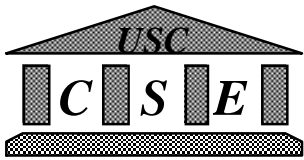
# Other Major COCOMO II Changes

- **Range versus point estimates**
- **Requirements Volatility replaced by Breakage %**
- **Multiplicative cost driver changes**
  - **Product CD's**
  - **Platform CD's**
  - **Personnel CD's**
  - **Project CD's**
- **Maintenance model and reuse model not identical**



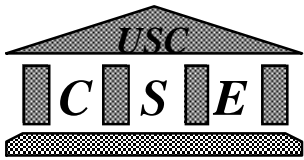
# Post-Architecture EMs - Product:

	Very Low	Low	Nominal	High	Very High	Extra High
Required Reliability (RELY)	slight inconvenience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
Database Size (DATA)		DB bytes/ Pgm SLOC < 10	10ŠD/P < 100	100ŠD/P < 1000	D/P • 1000	
Complexity (CPLX)	see Complexity Table					
Required Reuse (RUSE)		none	across project	across program	across product line	across multiple product lines
Documentation Match to Lifecycle (DOCU)	Many life cycle needs uncovered	Some life cycle needs uncovered	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	



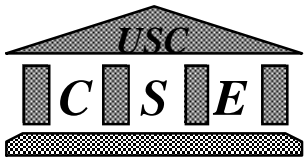
# Post-Architecture Complexity:

	<b>Control Operations</b>	<b>Computational Operations</b>	<b>Device - dependent Operations</b>	<b>Data Management Operations</b>	<b>User Interface Management Operations</b>
Very Low	...	...	...	...	...
Low	...	...	...	...	...
Nominal	Mostly simple nesting. Some intermodule control. Decision tables. Simple call backs or message passing, including middleware-supported distributed processing.	- Use of standard math and statistical routines. Basic matrix/vector operations.	I/O processing includes device selection, status checking and error processing.	Multi-file input and single file output. Simple structural changes, simple edits. Complex COTS-DB queries, updates.	Simple use of widget set.
High	...	...	...	...	...
Very High	...	...	...	...	...
Extra High	...	...	...	...	...



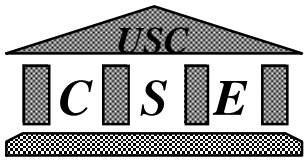
# Post-Architecture EMs - Platform:

	Very Low	Low	Nominal	High	Very High	Extra High
Execution Time Constraint (TIME)			≤50% use of available execution time	70%	85%	95%
Main Storage Constraint (STOR)			≤50% use of available storage	70%	85%	95%
Platform Volatility (PVOL)		major change every 12 mo.; minor change every 1 mo.	major: 6 mo.; minor: 2 wk.	major: 2 mo.; minor: 1 wk.	major: 2 wk.; minor: 2 days	



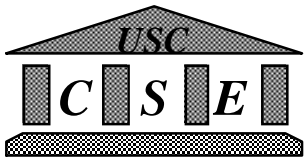
# Post-Architecture EMs - Personnel:

	Very Low	Low	Nominal	High	Very High	Extra High
Analyst Capability (ACAP)	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Programmer Capability (PCAP)	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Personnel Continuity (PCON)	48%/year	24%/year	12%/year	6%/year	3%/year	
Application Experience (AEXP)	Š 2 months	6 months	1 year	3 years	6 years	
Platform Experience (PEXP)	Š 2 months	6 months	1 year	3 years	6 years	
Language and Tool Experience (LTEX)	Š 2 months	6 months	1 year	3 years	6 years	



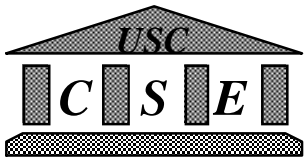
# Post-Architecture EMs - Project:

	<b>Very Low</b>	<b>Low</b>	<b>Nominal</b>	<b>High</b>	<b>Very High</b>	<b>Extra High</b>
Use of Software Tools (TOOL)	edit, code, debug	simple, frontend, backend CASE, little integration	basic lifecycle tools, moderately integrated	strong, mature lifecycle tools, moderately integrated	strong, mature, proactive lifecycle tools, well integrated with processes, methods, reuse	
Multisite Development: Collocation (SITE)	International	Multi-city and Multi - company	Multi-city or Multi - company	Same city or metro. area	Same building or complex	Fully collocated
Multisite Development: Communications (SITE)	Some phone, mail	Individual phone, FAX	Narrowband email	Wideband electronic communication	Wideband elect. comm, occasional video conf.	Interactive multimedia
Required Development Schedule (SCED)	75% of nominal	85%	100%	130%	160%	



# Early Design vs. Post-Arch EMs:

Early Design Cost Driver	Counterpart Combined Post Architecture Cost Drivers
Product Reliability and Complexity	RELY, DATA, CPLX, DOCU
Required Reuse	RUSE
Platform Difficulty	TIME, STOR, PVOL
Personnel Capability	ACAP, PCAP, PCON
Personnel Experience	AEXP, PEXP, LTEX
Facilities	TOOL, SITE
Schedule	SCED



# Other Model Refinements

- Initial Schedule Estimation

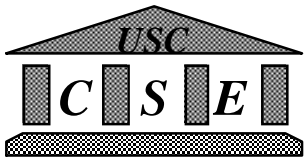
$$TDEV = \left[ 2.66 \times (\overline{PM})^{(0.33 + 0.2 \times (B - 1.01))} \right] \times \frac{SCED\%}{100}$$

where  $\overline{PM}$  = estimated person months excluding Schedule multiplier effects

- Output Ranges

Stage	Optimistic Estimate	Pessimistic Estimate
Application Composition	0.50 E	2.0 E
Early Design	0.67 E	1.5 E
Post-Architecture	0.80 E	1.25 E

- 80% confidence limits: 10% of time each below Optimistic, above Pessimistic
- Reflect sources of uncertainty in model inputs



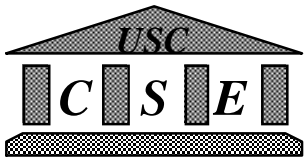
# COCOMO Model Comparisons

	COCOMO	Ada COCOMO	COCOMO II: Application Composition	COCOMO II: Early Design	COCOMO II: Post-Architecture
Size	Delivered Source Instructions (DSI) or Source Lines of Code (SLOC)	DSI or SLOC	Object Points	Function Points (FP) and Language	FP and Language or SLOC
Reuse	Equivalent SLOC = Linear $f(DM,CM,IM)$	Equivalent SLOC = Linear $f(DM,CM,IM)$	Implicit in Model	Equivalent SLOC = nonlinear $f(AA, SU, UNFM, DM, CM, IM)$	Equivalent SLOC = nonlinear $f(AA, SU, UNFM, DM, CM, IM)$
Breakage	Requirements Volatility rating: (RVOL)	RVOL rating	Implicit in Model	Breakage %: BRAK	BRAK
Maintenance	Annual Change Traffic (ACT) = %added + %modified	ACT	Object Point ACT	$f(ACT, SU, UNFM)$	$f(ACT, SU, UNFM)$
Scale (b) in $MM_{NOM}=a(Size)^b$	Organic: 1.05 Semidetached: 1.12 Embedded: 1.20	Embedded: 1.04 -1.24 depending on degree of: <ul style="list-style-type: none"> <li>early risk elimination</li> <li>solid architecture</li> <li>stable requirements</li> <li>Ada process maturity</li> </ul>	1.0	1.0-1.26 depending on the degree of: <ul style="list-style-type: none"> <li>precedentedness</li> <li>conformity</li> <li>early architecture, risk resolution</li> <li>team cohesion</li> <li>process maturity (SEI)</li> </ul>	1.0-1.26 depending on the degree of: <ul style="list-style-type: none"> <li>precedentedness</li> <li>conformity</li> <li>early architecture, risk resolution</li> <li>team cohesion</li> <li>process maturity (SEI)</li> </ul>
Product Cost Drivers	RELY, DATA, CPLX	RELY*, DATA, CPLX*, RUSE	None	RCPX**†, RUSE**†	RELY, DATA, DOCU**†, CPLX†, RUSE**†
Platform Cost Drivers	TIME, STOR, VIRT, TURN	TIME, STOR, VMVH, VMVT, TURN	None	Platform difficulty: PDIF**†	TIME, STOR, PVOL(=VIRT)
Personnel Cost Drivers	ACAP, AEXP, PCAP, VEXP, LEXP	ACAP*, AEXP, PCAP*, VEXP, LEXP*	None	Personnel capability and experience: PERS**†, PREX**†	ACAP*, AEXP†, PCAP*, PEXP**†, LTEX**†, PCON**†
Project Cost Drivers	MODP, TOOL, SCED	MODP*, TOOL*, SCED, SECU	None	SCED, FCIL**†	TOOL**†, SCED, SITE**†

\* Different Multipliers

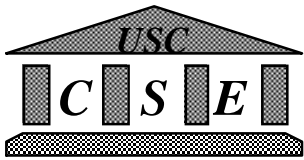
† Different Rating Scale

9/21/98



# Outline

- **COCOMO II overview**
  - Objectives and strategy
  - Model specifics
- ➔ • **COCOMO II usage example**
- **COCOMO II experience and plans**
  - Model calibration
  - Usage experience and plans
- **Model-Based Architecting and Software Engineering**



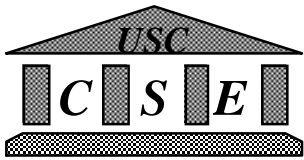
# Library SDI Description

## 1. LSDI Architecture Summary

Socal University (SCU) has three major campuses, each of which has a main library which will provide LSDI services. Each of three campuses operates a server running a COTS client-server library services package. The COTS package provides Internet capabilities enabling a client at any campus to deal with any of three servers.

The SCU Library, Computer Sciences Department, and Computing Services Operation have been funded to develop an experimental Selective Dissemination of Information (SDI) system to provide SCU users with information about new library acquisitions of interest. It will do this by comparing attributes of new library acquisitions with interest profiles provided by library users. The grant provides \$850K to develop the system.

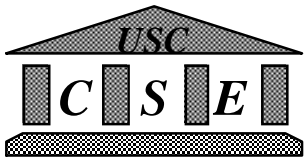
The SDI software will acquire data on recent library acquisitions, compare it with library user interest profiles, generate user notices, and support query, browsing, and request functions for the new acquisitions. It will have a number of subsidiary functions for user interest profile management, access control, user interface functions, and usage monitoring. It will communicate with users via the COTS client-server package.



## 2. Candidate SDI Software Functions and Cost Driver Ratings

### Candidate Software Functions

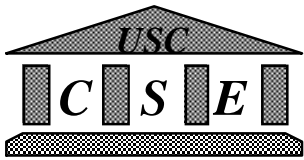
- **User Interest Profile Management (5 KDSI).** User interest profile creation, deletion, update, query, etc.
- **Access Control**
  - **Basic (4 KDSI).** Basic password access control.
  - **Extended (8 KDSI).** Basic plus authentication, authorization, instruction deletion.
  - **Rigorous (15 KDSI).** Extended plus formal specification and verification.
- **Acquisition Data Handling and Profile Checking:** acquisition record and file access, validity checking, logging, storage, etc. Determination of matches between acquisition and interest profiles.
  - **Basic Service (3 KDSI).** simple matching; standard library categories.
  - **Extended Service (6 KDSI).** Conditional matching; standard plus locally-defined categories.
- **User Interface and Services**
  - **Basic Service (7 KDSI).** Acquisition-match query, browse, and request functions. Basic help and profile management functions.
  - **Extended Service (12 KDSI).** Basic Service plus tutorial functions, interoperability with COTS user interface.



- **Usage Monitoring (4 KDSI).** Recording and summarizing system usage by time of day/week/year, by use attributes, by acquisition attributes.
- **Trend Analysis (3 KDSI)** Extrapolation of rates of increase, decrease, or cyclical trends in services.
- **Library network access (6 KDSI).** Extension of acquisition notification to cover acquisitions from a regional network of 10 additional universities.
- **COTS integration.** Software needed to integrate SDI functions with COTS library information functions:
  - 5 KDSI for processor X;
  - 8 KDSI for processor Y.

COCOMO II software development and cost drivers

**Some of the COCOMO II cost drivers are associated with hardware options to be specified in the next section. Below are the ratings determined for the remainder of the LSDI system cost drivers. Most of them are the same for all of the candidate software functions; any differences are identified on the following page**



## Effort Multipliers

**RELY Nominal, except for  
Access Control, which is  
High**

**DATA High**

**CPLX Nominal, except for  
Access Control (High) and  
COTS Integration (High)**

**RUSE Nominal**

**DOCU Nominal**

**STOR Nominal**

**PVOL Low**

**ACAP Nominal**

**PCAP Nominal**

**AEXP Nominal**

**PCON Nominal**

**LTEX High**

**SITE Very High**

**SCED Nominal**

**Scale Factors**

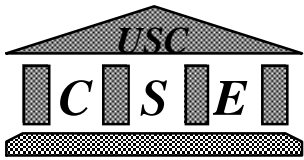
**PREC Nominal**

**FLEX High**

**RESL Nominal**

**TEAM Very High**

**PMAT Nominal**



### 3. Hardware Options

There are two hardware options for SDI functions. Processor X is a more mature but slower processor, with better tool support. Processor Y is a fast, new processor with a lower level of tool support and virtual machine experience. A summary of their comparative cost/performance/risk feature is as follows:

	<b>Processor X</b>	<b>Processor Y</b>
<b>COCOMO TIME rating</b>	<b>High</b>	<b>Nominal</b>
<b>COCOMO PEXP rating</b>	<b>Nominal</b>	<b>Low</b>
<b>COCOMO TOOL rating</b>	<b>High</b>	<b>Nominal</b>
<b>Nominal response time</b>	<b>5 sec</b>	<b>1 sec</b>
<b>System development risk</b>	<b>Low</b>	<b>High</b>
<b>Hardware cost</b>	<b>\$12K</b>	<b>\$30K</b>