

Course Overview

CS 577a/591a

Software Engineering I

Barry Boehm, USC



Outline

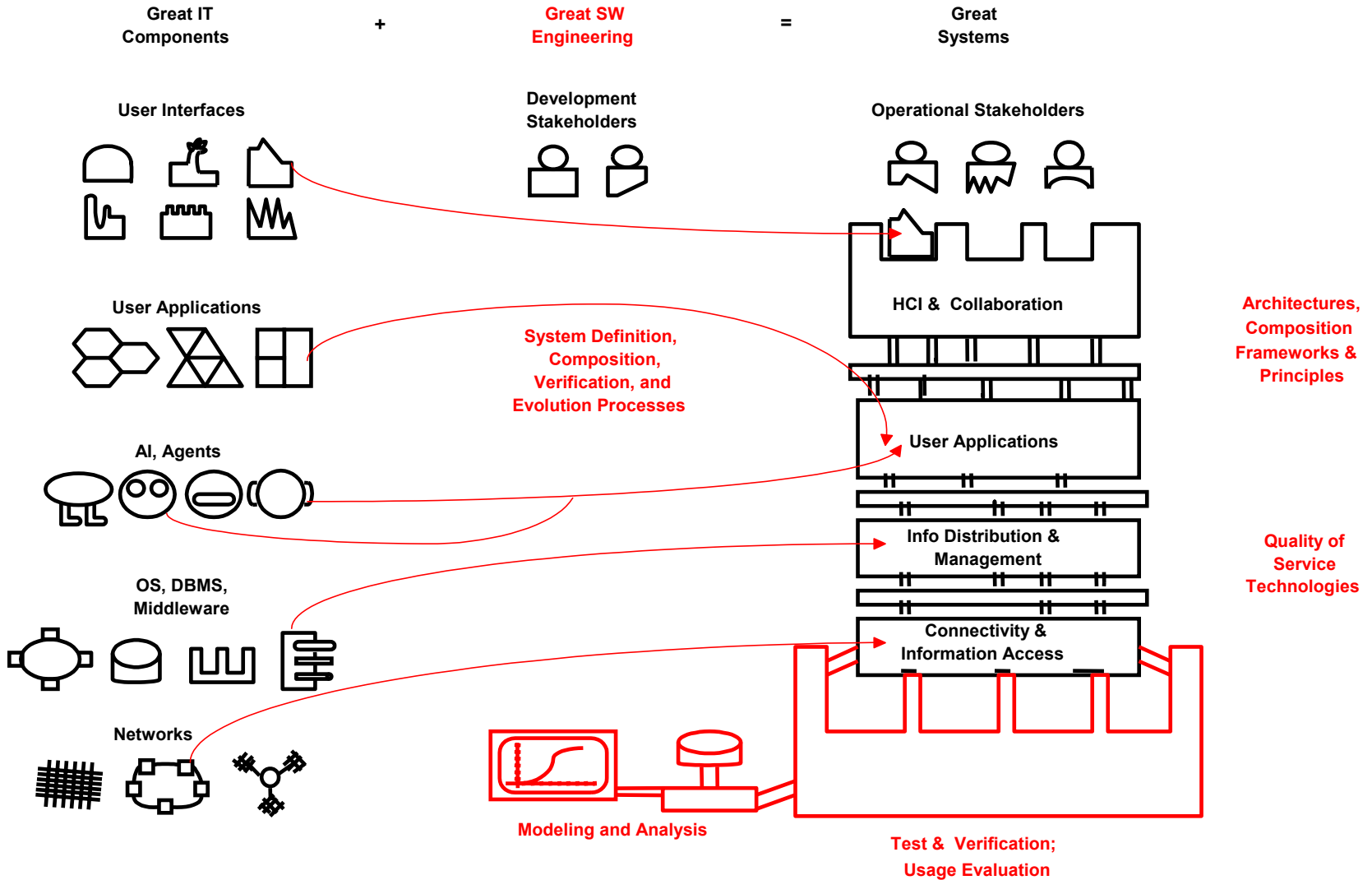
- **Software Engineering Definition and Learning Objectives**
 - Comparison of CS 510 and CS577a
- **Class Overview**
 - Class Project Objectives and Approach
 - Top 10 Risk Items
- **Model Clashes and MBASE**
- **Uni Word Case Study (Risk and Model Clash Analysis)**
 - Due Wed Sept 5 (start of class). Bulleted lists ok

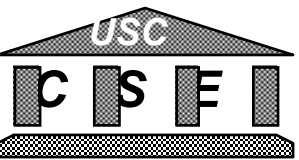
“ Software Engineering:” The disciplines which distinguish the coding of a computer program from the development of a software product.

Issues \ Stages	Requirements, Architecture	Design, Code	Test, Implement, Maintain
Computer Science		CS Focus	
User Applications			
Economics			
People			

- **Accommodate new tools and techniques**
 - **Web browsers, GUI prototypers, WinWin, Spiral processes**
- **Integrate all these considerations**
 - **Via integrated models (MBASE)**

Role of Software Engineering in IT Research and Systems





Main Software Engineering Challenges

- What to build? Why? How well?
 - Stakeholder needs balancing; business case
- Whom to build it with? Where?
 - Staffing; organizing; outsourcing
- How to build? When; in what order?
 - Process, methods, tools, components, increments
- How to adapt to change?
 - In user needs, technology, marketplace
- How much is enough?
 - Functionality, quality, specifying, prototyping, test



Software Technology Trends: Languages

Software Technology

Microprogramming

Low- level language programming

High- level language programming

Object-based and object-oriented programming

Component based development

- Domain architecture
- Reusable components
- Domain languages



Language Level

Bits: 100, 010
F12, A07, 124, AAF



Instructions: LDR, ADDX,
CLA, JMPS



Lines: If A then B
loop
I = I + 1



Objects and packages:
Type color is (red, yellow, green);
package traffic_light
when green go;



Components and services:

Overlay map with grid;
When failure switchover;
Shutdown all test processes;

Point & click equivalents

Support Software

Machine languages

Assemblers
Linkers

Compilers
Operating systems

Compilers
Operating systems
Runtime libraries

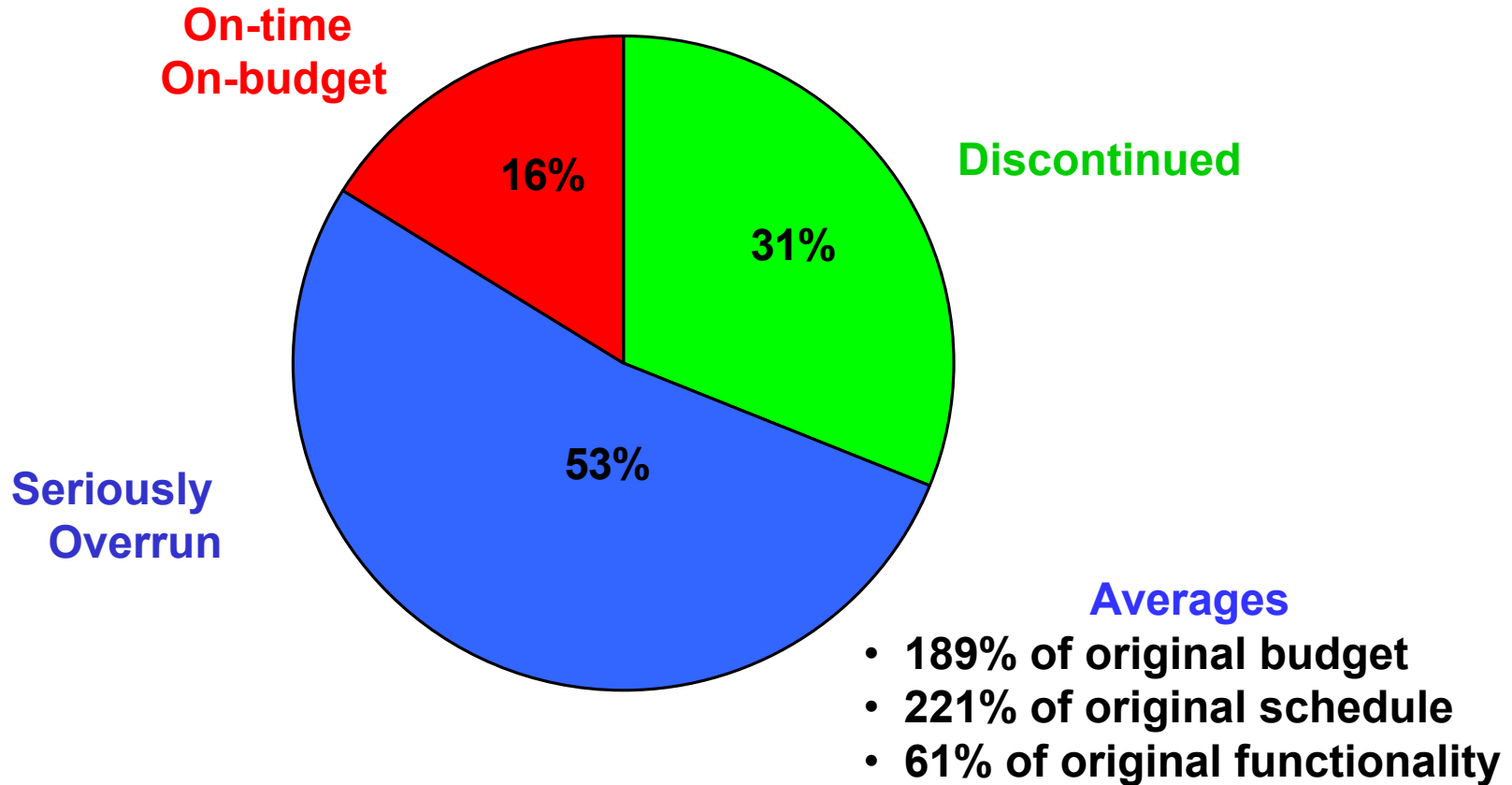
Compilers
Operating systems
Runtime libraries
Networks
Middleware
CASE tools



A Short History of Software Processes - necessarily oversimplified

Decade	Orientation	Example(s)	Problems
1950's	Engineering	SAGE	Engineering orientation
1960's	Programming	Code and fix	Rework, scalability
1970's	Requirements	Waterfall	Risk, GUI, COTS/reuse
1980's	Many	Evolutionary, Incremental, Spiral, Helix, JAD, RAD	Overgeneralization/ Overspecialization
1990's	Emergence	Win-Win Spiral, Rational, Adaptive (model generators)	Value and economics
2000's	Value	Benefits realization, MBASE	Integration with emergence

Software Engineering Is Not Well-Practiced Today -Standish Group CHAOS Report 1995





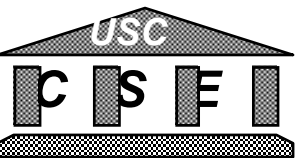
Information Technology Trends

Traditional Development

- Standalone systems
- Stable requirements
- Rqts. determine capabilities
- Control over evolution
- Enough time to keep stable
- Repeatability-oriented process, maturity models

Current/Future Trends

- Everything connected (maybe)
- Rapid requirements change
- COTS capabilities determine rqts.
- No control over COTS evolution
- Ever-decreasing cycle times
- Adaptive process models



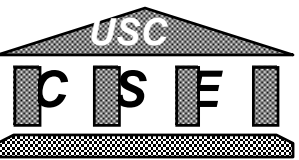
Traditional vs. Emerging SW Processes

Traditional

- **Contract-oriented**
 - Your problem; win-lose
- **Sequential**
- **Procedure-driven**
- **Programming-driven**
- **Many interlocking milestones**
- **Focus on discipline & control**

Emerging

- **Collaboration-oriented**
 - Our problem; win-win
- **Cyclic, concurrent**
- **Risk-driven**
- **Reuse-driven**
- **Critical anchor points with subsidiary milestones**
- **Mix of flexibility & discipline**



Comparison of CS 510 and CS 577a/591a

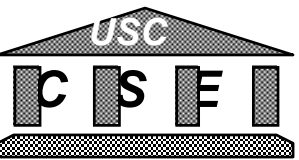
CS 510

- **COCOMO II Extensions**
- **Microeconomics**
 - Decision Theory
- **Rapid Application Development**
- **People Management**
- **Term Paper**

- **MBASE**
- **WinWin Spiral**
 - Risk Management
- **Planning & Control**
 - COCOMO II
- **Business Case Analysis**

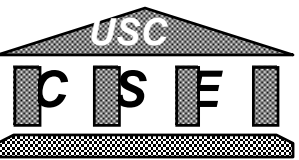
CS 577a/ 591a

- **S/W - System Architecting**
- **Operational Concept & Rqts. Definition**
 - WinWin System
 - Prototyping
- **OO Analysis & Design**
 - Rational Rose
- **Team Project (591a: IV&V)**



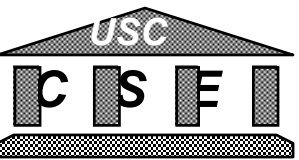
Class Project Objectives

- **Create life-cycle architecture and plan for developing an**
- **E-Services capability for the**
- **USC Information Services Division**
 - plus a few supplementary projects



Class Project Artifacts

- 1. An Operational Concept Definition**
- 2. A Prototype of Key System Features**
- 3 A System Requirements Definition**
- 4. A System and Software Architecture Definition**
- 5. A Life Cycle Plan**
- 6. A Feasibility Rationale, assuring the consistency and feasibility of items 1-5**



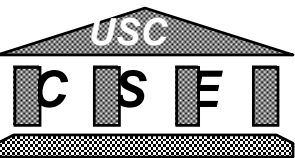
Class Project Approach

- **Artifacts developed concurrently**
 - Using Win-Win Spiral approach
- **Two critical project milestones (anchor points)**
 - Life Cycle Objectives (LCO)
 - Life Cycle Architecture (LCA)



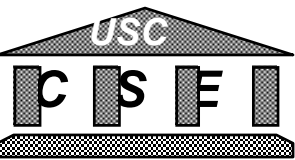
Team Structure

- **Five-person teams**
 - Ops Concept and Requirements team members also develop Life Cycle Plan and Quality Plan
 - **Feasibility Rationale member acts as Project Manager**
 1. Ensures consistency among the team members' artifacts (and documents this in the Rationale).
 2. Leads the team's development of plans for achieving the project results, and ensures that project performance tracks the plans.
- Teams formed by Monday, Sept. 10**
- Web questionnaires should help in team formation



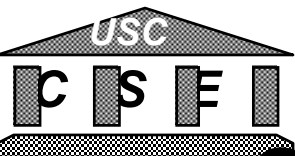
WinWin User Negotiations

- **Teams work with representatives of USC e-services users**
 - Art, cinema, engineering, business, etc.
 - E-services infrastructure
 - Begin with problem statements of top digital library needs
- **Use Easy WinWin system to balance user needs with customer and developer win conditions**



Major Class Project Milestones

September 10	--	All teams formed
September 21	--	Initial Prototypes
October 1	--	WinWin Negotiation Results
October 22	--	LCO Drafts on Web Site
Oct. 29-Nov. 2	--	LCO Architecture Reviews
November 5	--	LCO Package Due
November 28	--	LCA Drafts on Web Site
December 3-7	--	LCA Architecture Reviews
December 10	--	LCA Package Due
December 12	--	Individual Critiques Due



CS 577a, Fall 2001 Questionnaire

Please fill out and return.

Name: _____

Student ID #: _____

Dept./Degree Program: _____

Job, Employer: _____

Software Work Experience (years): _____

Phone, fax numbers: _____

E-mail Address: _____

Acknowledgement: I acknowledge the importance of USC's academic integrity standards (with respect to plagiarism, referencing others' work, etc.), and agree to abide by them.

Signature: _____

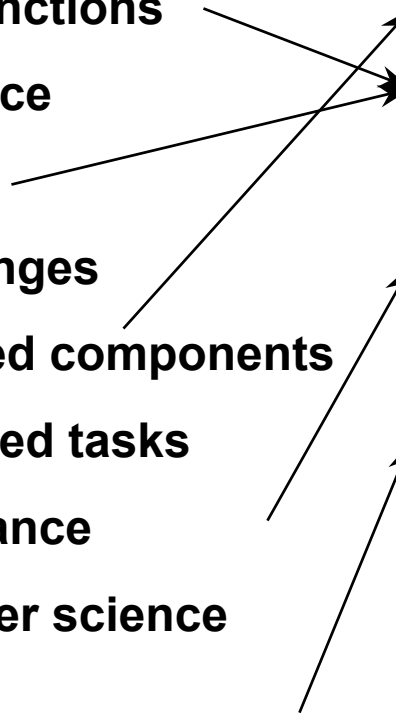
Top 10 Risk Items: 1989 and 1995

1989

1. Personnel shortfalls
2. Schedules and budgets
3. Wrong software functions
4. Wrong user interface
5. Gold plating
6. Requirements changes
7. Externally-furnished components
8. Externally-performed tasks
9. Real-time performance
10. Straining computer science

1995

1. Personnel shortfalls
2. Schedules, budgets, process
3. COTS, external components
4. Requirements mismatch
5. User interface mismatch
6. Architecture, performance, quality
7. Requirements changes
8. Legacy software
9. Externally-performed tasks
10. Straining computer science





Primary CS577 Risk Items

- **Personnel: commitment; compatibility; ease of communication; skills (management, Web/Java, Perl, CGI, data compression, ...)**
- **Schedule: project scope; IOC content; critical-path items (COTS, platforms, reviews, ...)**
- **COTS: see next chart; multi-COTS**
- **Rqts, UI: mismatch to Library user needs**
- **Performance: #bits; #bits/sec; overhead sources**
- **External tasks: Client/Operator preparation, commitment for transition**



COTS and External Component Risks

- **COTS risks: immaturity; inexperience; COTS incompatibility with application, platform, other COTS; controllability**
- **Non-commercial off-the shelf components: reuse libraries, government, universities, etc.**
 - **Qualification testing; benchmarking; inspections; reference checking; compatibility analysis**



Outline

- **Software Engineering Definition and Learning Objectives**
 - Comparison of CS 510 and CS577a
- **Class Overview**
 - Class Project Objectives and Approach
 - Top 10 Risk Items
- ➔ • **Model Clashes and MBASE**
- **Uni Word Case Study (Risk and Model Clash Analysis)**
 - Due Wed Sept 5 (start of class). Bulleted lists ok

“No scene from prehistory is quite so vivid as that of the mortal struggles of great beasts in the tar pits.



Large system programming has over the past decade been such a tar pit, and many great and powerful beasts have thrashed violently in it.”

Fred Brooks, 1975

“Everyone seems to have been surprised by the stickiness of the problem, and it is hard to discern the nature of it.



But we must try to understand it if we are to solve it.”

Fred Brooks, 1975

Understanding the Tar Pit: Model Clashes

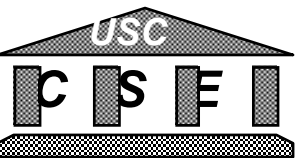


- **Model Clash: An incompatibility among the underlying assumptions of a set of models**
 - Process, product, property, and success models
 - Often unrecognized
 - Produces conflicts, confusion, mistrust, frustration, rework, throwaway systems



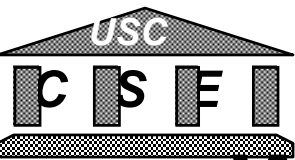
Examples of Model Clashes

- **Design-to-schedule process, unprioritized requirements, and tightly-coupled architecture**
- **COTS-driven product and Waterfall process**
- **Risk-based process and spec-based progress payments**
- **Evolutionary development without life-cycle architecture**
- **Spec-based process and IKIWISI success model**
 - I'll know it when I see it
- **Golden Rule and stakeholder win-win**



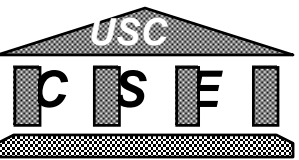
The Golden Rule as Software Success Model

- **Do unto others**
- **As you would have others do unto you**
- **Build computer systems to serve users and operators**
- **Assuming users and operators like to write programs, and know computer science**
- **Computer science world (Compilers, OS, etc.)**
 - Users love powerful, obscure, UNIX-like commands
- **Applications world**
 - Users are pilots, doctors, librarians: Keep it simple
- **Better to use Platinum Rule**
 - Do unto others as they would be done unto

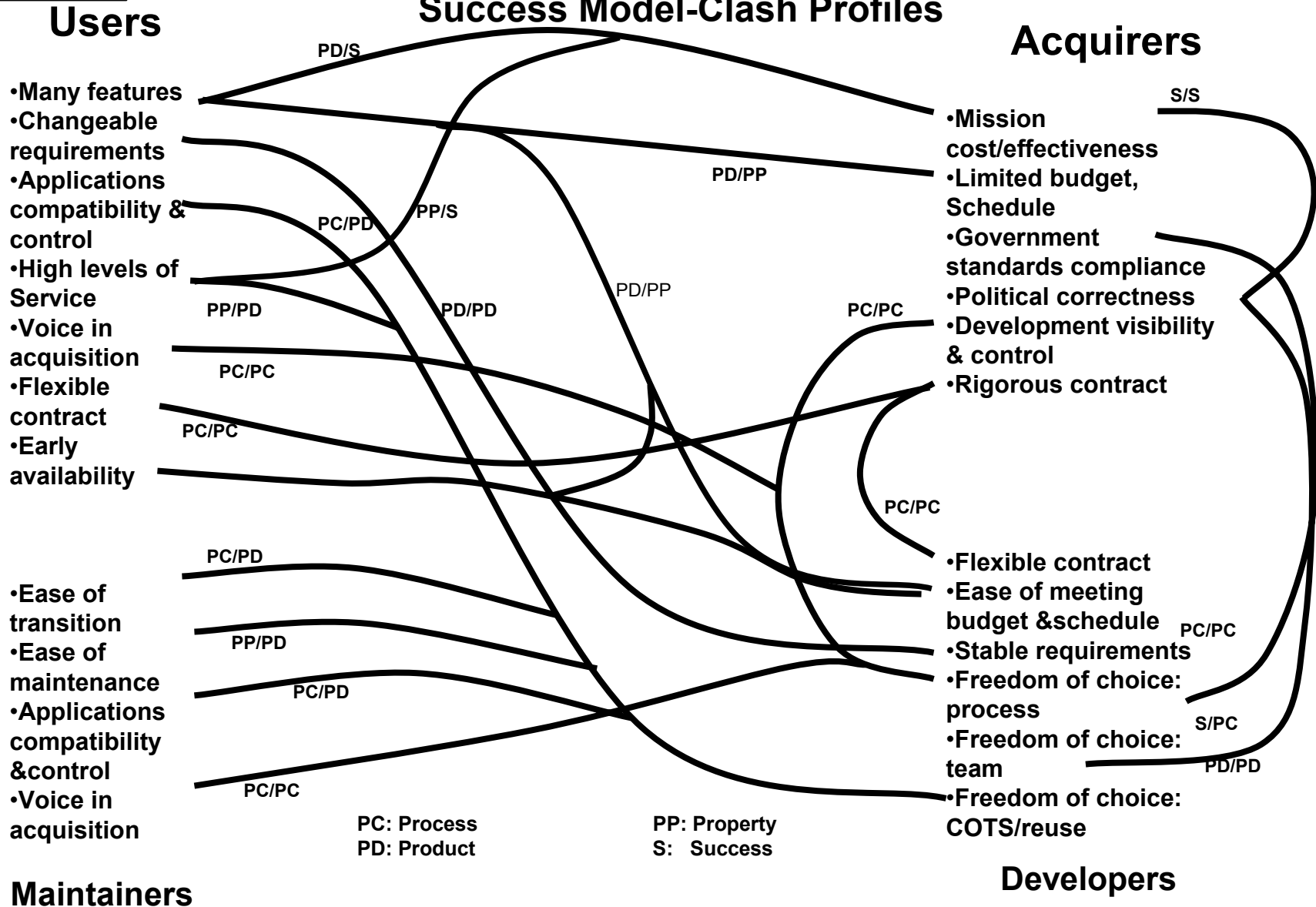


Where do Models (and Clashes) Come From?

- **Childhood training**
 - Golden Rule, easiest - first
- **Past experience**
 - Waterfall, Add people to speed up
- **Exaggerating for effect**
 - Quality is free, COTS marketing
- **Government/Corporate policy**
 - Use waterfall, use COTS, use Ada, use 4GL's,
Cost as Independent Variable
- **Multi-Stakeholder success model clashes**

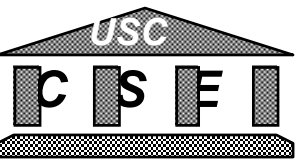


Success Model-Clash Profiles

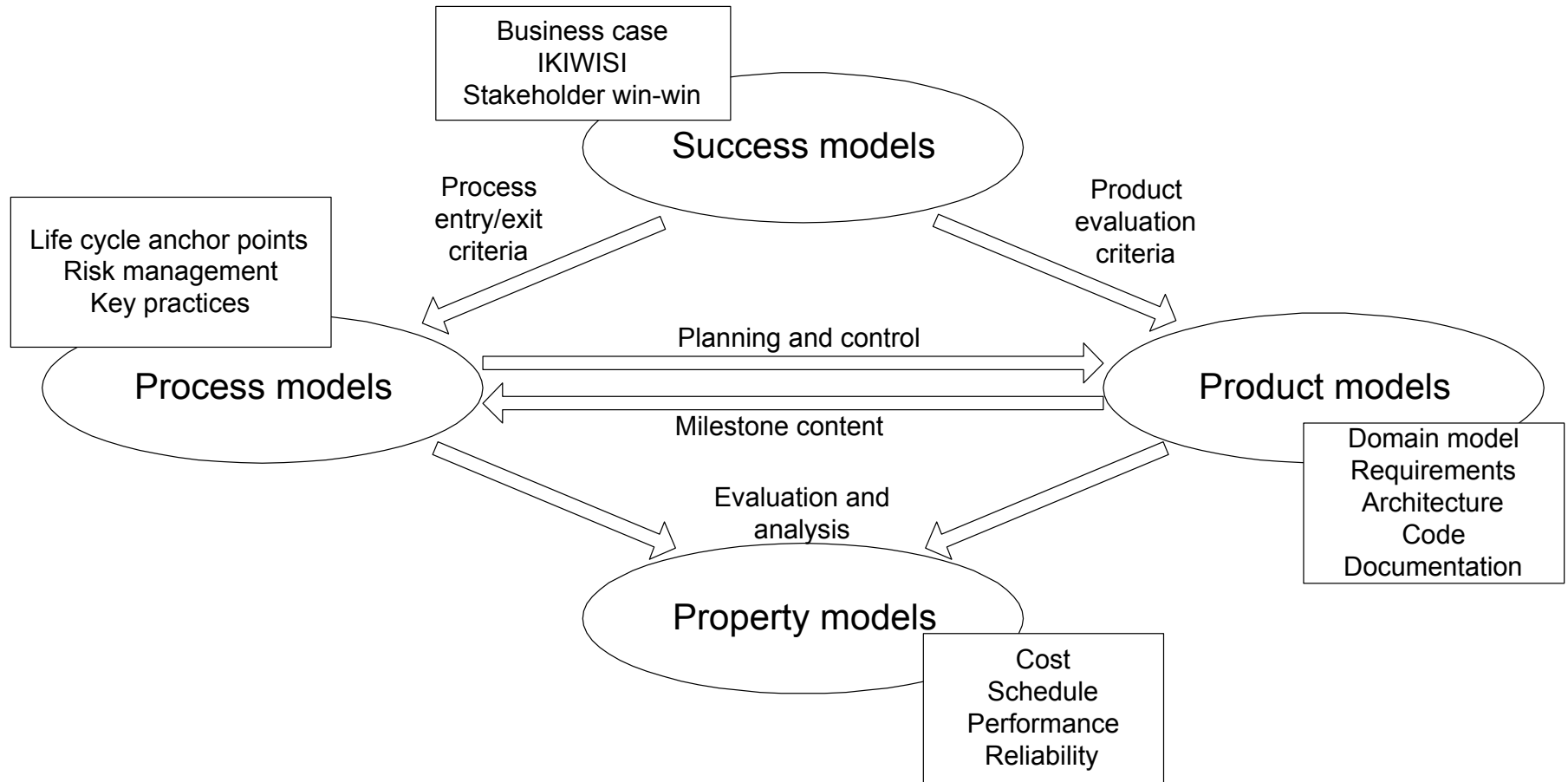


Clashes Among MBASE Models

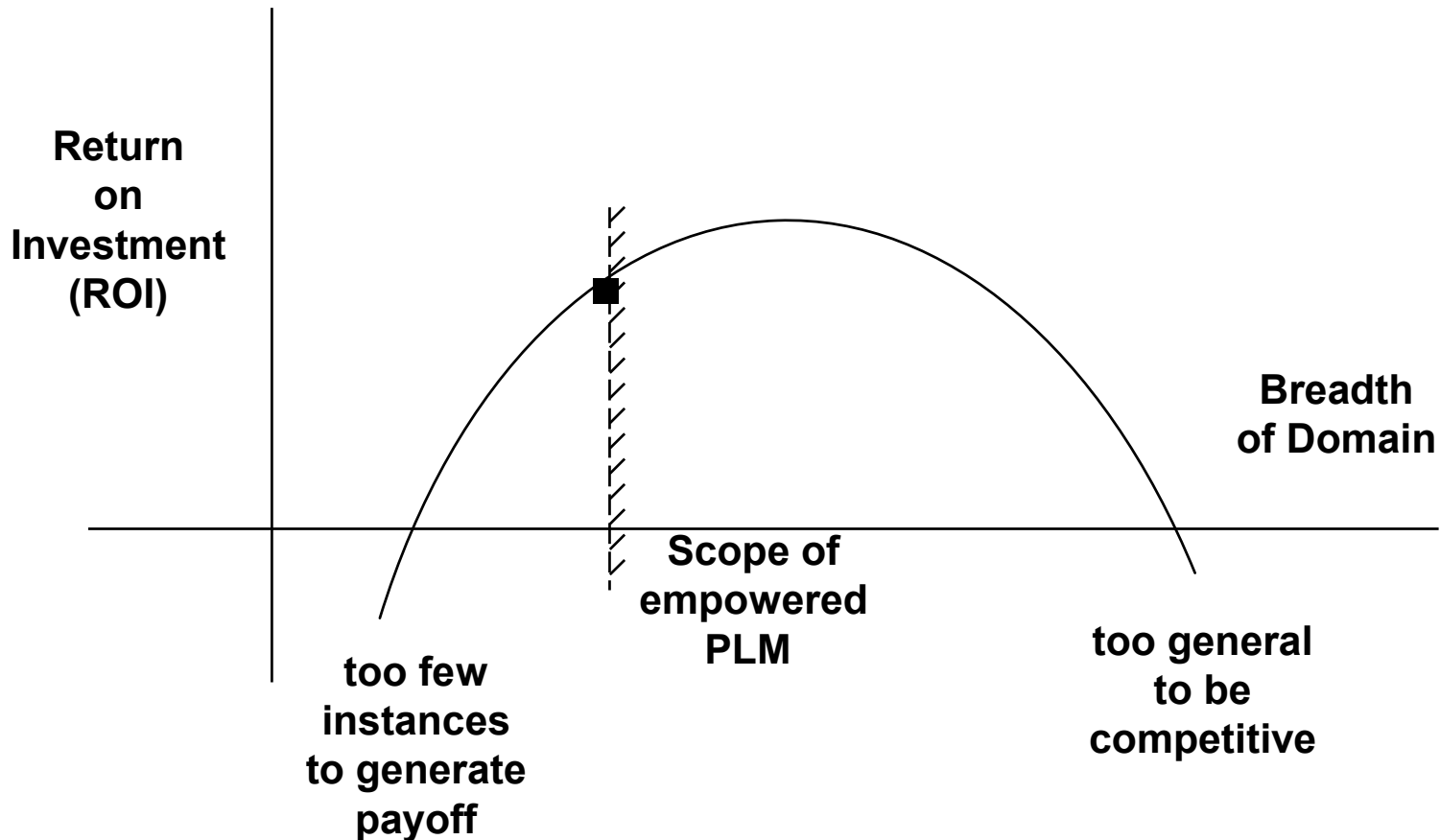
	Product Model	Process Model	Property Model	Success Model
Product Model	<ul style="list-style-type: none"> • Structure clash • Traceability clash • Architecture style clash 	<ul style="list-style-type: none"> • COTS-driven product vs. Waterfall (requirements-driven) process 	<ul style="list-style-type: none"> • Interdependent multiprocessor product vs. linear performance scalability model 	<ul style="list-style-type: none"> • 4GL-based product vs. low development cost and performance scalability
Process Model		<ul style="list-style-type: none"> • Multi-increment development process vs. single-increment support tools 	<ul style="list-style-type: none"> • Evolutionary development process vs. Rayleigh-curve cost model 	<ul style="list-style-type: none"> • Waterfall process model vs. "I'll know it when I see it" (IKIWISI) prototyping success model
Property Model			<ul style="list-style-type: none"> • Minimize cost and schedule vs. maximize quality (Quality is free) 	<ul style="list-style-type: none"> • Fixed-price contract vs. easy-to-change, volatile requirements
Success Model				<ul style="list-style-type: none"> • Golden Rule vs. stakeholder win-win

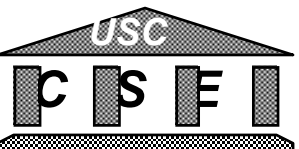


MBASE Integration Framework

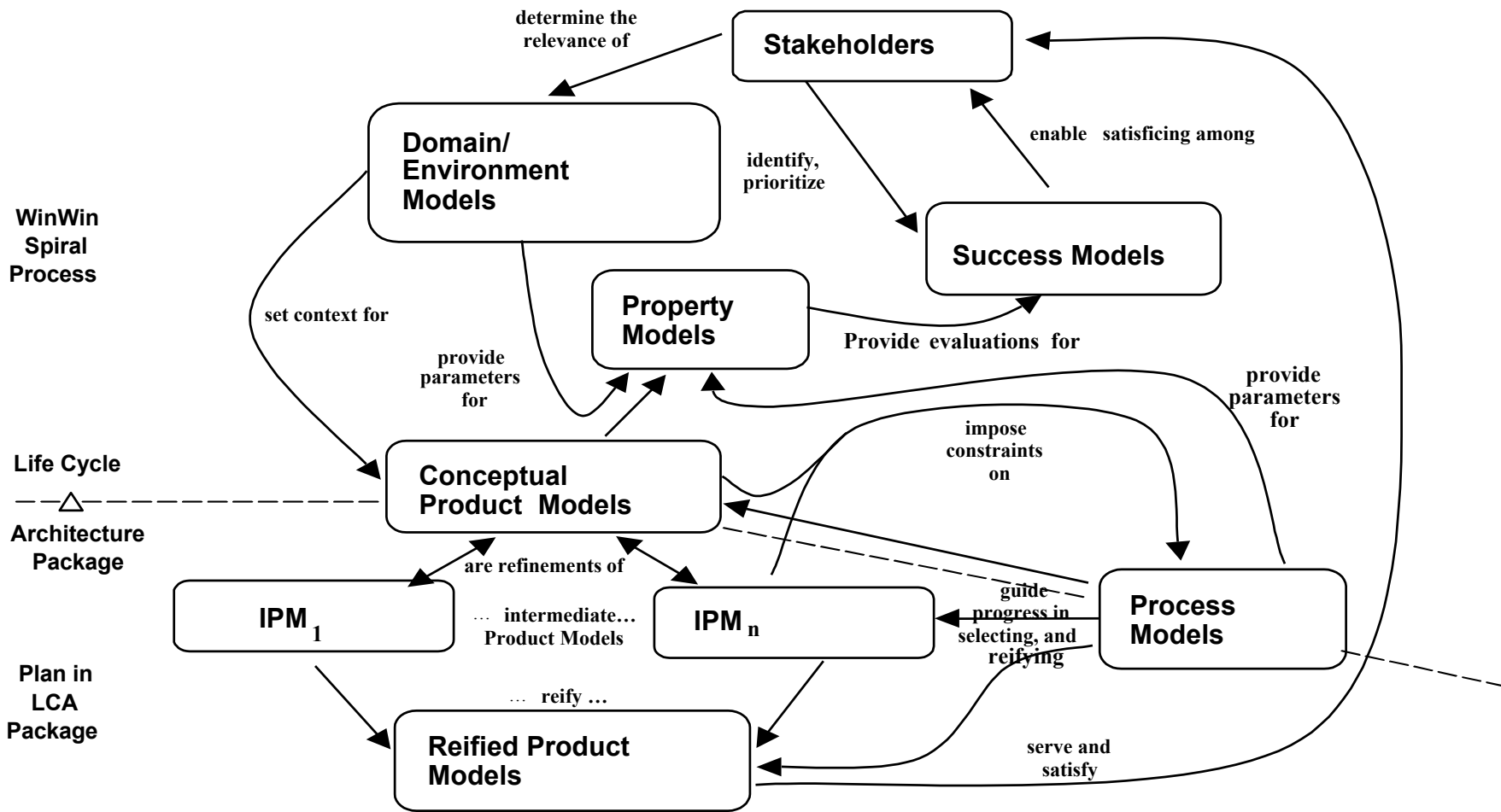


Product Line Domain Scope a Function of ROI, Scope of Empowered PL Manager



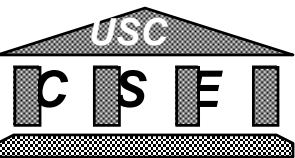


MBASE Conceptual Framework



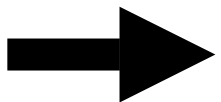
Success Models Drive Other Model Choices

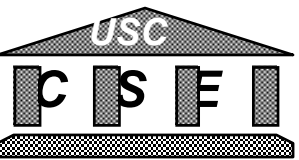
Success Model	Demo agent-based E-commerce system at COMDEX in 9 months	Safe air traffic control system
Key Stakeholders	Entrepreneurs, venture capitalists, customers	Controllers, Govt. agencies, developers
Key Property Models	Schedule estimation	Safety models
Process Model	Design-to-schedule	Initial spiral to risk-manage COTS, etc.; Final waterfall to verify safety provisions
Product Model	Domain constrained by schedule; architected for ease in dropping features to meet schedule	Architected for fault tolerance, ease of safety verification



Outline

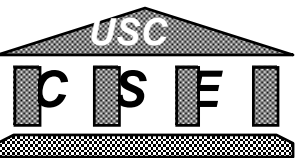
- **Software Engineering Definition and Learning Objectives**
 - Comparison of CS 510 and CS577a
- **Class Overview**
 - Class Project Objectives and Approach
 - Top 10 Risk Items
- **Model Clashes and MBASE**
- **Uni Word Case Study (Risk and Model Clash Analysis)**
 - Due Wed Sept 5 (start of class). Bulleted lists ok





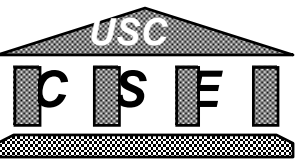
UniWord Case Study

- **Background**
- **The Competition (7-9/83)**
- **Project Startup (9-10/83)**
- **Early Problems 10/83-2/84)**
- **Problem Interactions (2-4/84)**
- **The Rescue Attempt (4-5/84)**
- **The Assessment (5-6/84)**



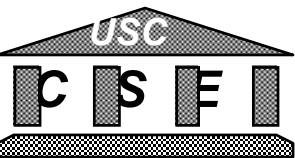
Background

- **Universal Micros, Inc. Developing UniWindow Workstation**
 - Two 32-Bit, 15-MHz Micros
 - 1 Mbyte main Memory
 - 20 Mbyte Winchester Disk Drive
 - High-Resolution, Bit-Map, Color Display
 - UniWindow Operating System(based on Unix)
- **Target: Demo at NCC, July 1984**
- **Contracting for UniWord Word Processing System**
 - High-Level Functional Specs Developed
 - 9-Month Period of Performance



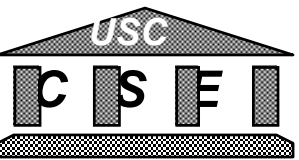
The Competition

- **5 Responses to RFP**
 - 2 Unqualified
 - 1 Too Expensive
 - Soft Wizards, Inc.
 - Text Products, Inc.
- **Final 2 Asked for Best and Final Offer**
- **Soft Wizards Added Functions, Cut Price \$100K**
 - Get Leading position in Word Processing Software For Advanced Workstations
 - Needed Income
 - Make up Deficit With License Fees
- **Job Awarded to Soft Wizards 15 Sept 1983**



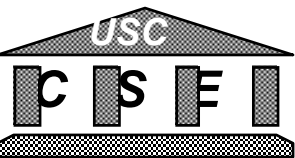
Project Startup (Month 1)

- **Project Manager: Bill Brown**
 - CP/M-Based Text Editor
- **Deputy PM: Karen Gray**
 - UNIX, VI Expert
- **Major Differences on Text Editor**
 - Brown: Simple, Menu Oriented
 - Gray: Complex, Command Oriented
- **1 Month to Resolve This Issue**
 - Gray Develop Text Editor
 - Brown Lead Development of Other Components



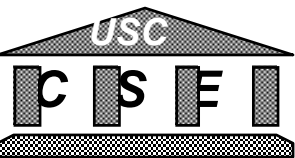
Early Problems (Months 2-5)

- **Integration Started January 1984**
 - No Test Drivers, Test Data
 - 3 Components: Text Editor, File Manager, Format/Runoff
- **Interface Incompatibilities**
 - File Structures, Buffers, Window Controls
- **Attempts to Compromise Fail**
 - Karen Gray Leaves Company
- **File Manager Progress Slow**
 - Sam Silver, DBMS Expert, Put On Job



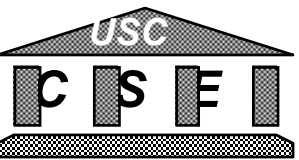
Problems Interactions (Months 6-7)

- **Hard to modify Text Editor**
 - Lave It Alone, Build Layer Around It
- **Silver Begins Building Relational DBMS**
 - Suggested by Universal Micros Marketer
 - No Formal Customer Request
- **Integration Problems Deepen**
 - Proliferation of Program Versions
 - No Interface Specs
 - Major Drain on Developers' Time
- **Brown Requests Four Additional people to Help With Integration for 6 Weeks**
 - Request Granted



The Rescue Attempt (Month 8)

- **New People Experienced, But not With UniWord**
- **Documentation Scarce, Out of Date**
- **Developers Required to Explain System to New People**
 - Progress Slows Down Further
- **Conclusion: Delivery Date Can't Be Achieved**



Project Assessment (Month 9)

- **Soft Wizard/Universal Micro Meeting**
 - **Text Editor Demo: Unsatisfactory**
 - **Poor match to UniWindow Strengths**
 - **Summary of U.M. Change Requests**
 - **Could Deliver Legally Acceptable Product, But Nobody would Like It**
 - **Requested Another \$100K, 3 Months to Complete**
- **Universal Micros' Response**
 - **Drop UniWord from July NCC Demo**
 - **Hire Consultant to Audit Project**



Homework Assignment #1

Due Wednesday, Sept. 5, 30 points

- **Read the UniWord case study (Web: Electronic Papers)**
- **Identify project risk items that could have been better addressed by either Universal Micros or Soft Wizards.**
- **Identify model clashes between process, product, property, and success models**
- **Bulleted lists OK.**