



# **Mastering Rapid Delivery and Change with the SAIV Process Model - Schedule As Independent Variable**

**Barry Boehm, USC**

**ESCOM 2001**

**April 4, 2001**

**<http://sunset.usc.edu>**



# Outline

- **Motivation**
- **The MBASE Process Framework**
  - **Model-Based (System) Architecting and Software Engineering**
  - **E-Commerce SAIV example**
- **The SAIV Process Model**
  - **Usage experience**
- **Conclusions**



# Is This A Problem?

- **The clients want this system on a ridiculously short schedule**
- **And they aren't even willing to commit on just exactly what they want**



# Is This A Problem?

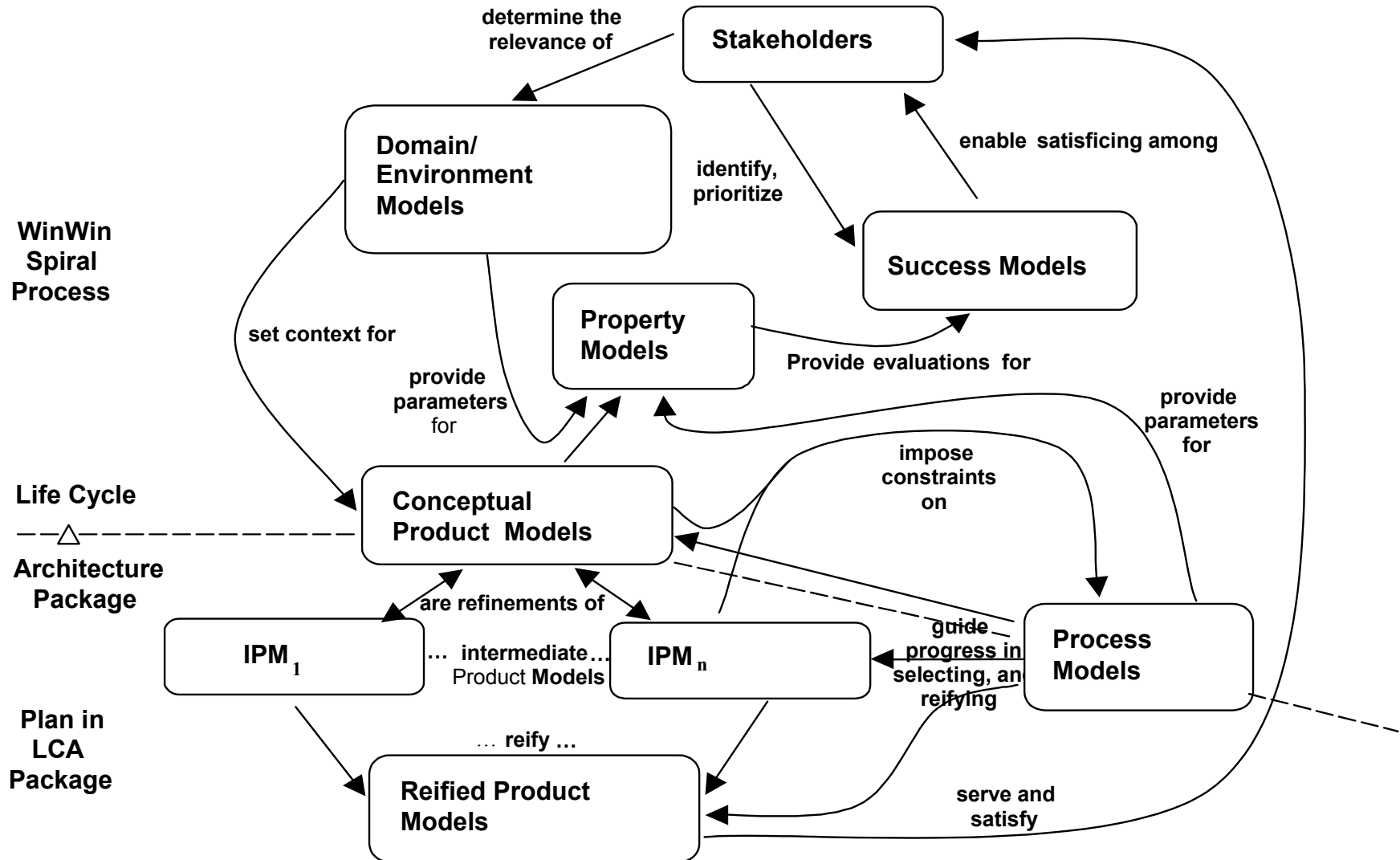
- **The clients want this system on a ridiculously short schedule**
- **And they aren't even willing to commit on just exactly what they want**

**Usually, it leads to a Death March**

# Is This A Problem?

- **The clients want this system on a ridiculously short schedule**
  - **And they aren't even willing to commit on just exactly what they want**
- Usually, it leads to a Death March, but...**
- **Usually, it's an opportunity to succeed via the SAIV process model**
    - **Schedule As Independent Variable**

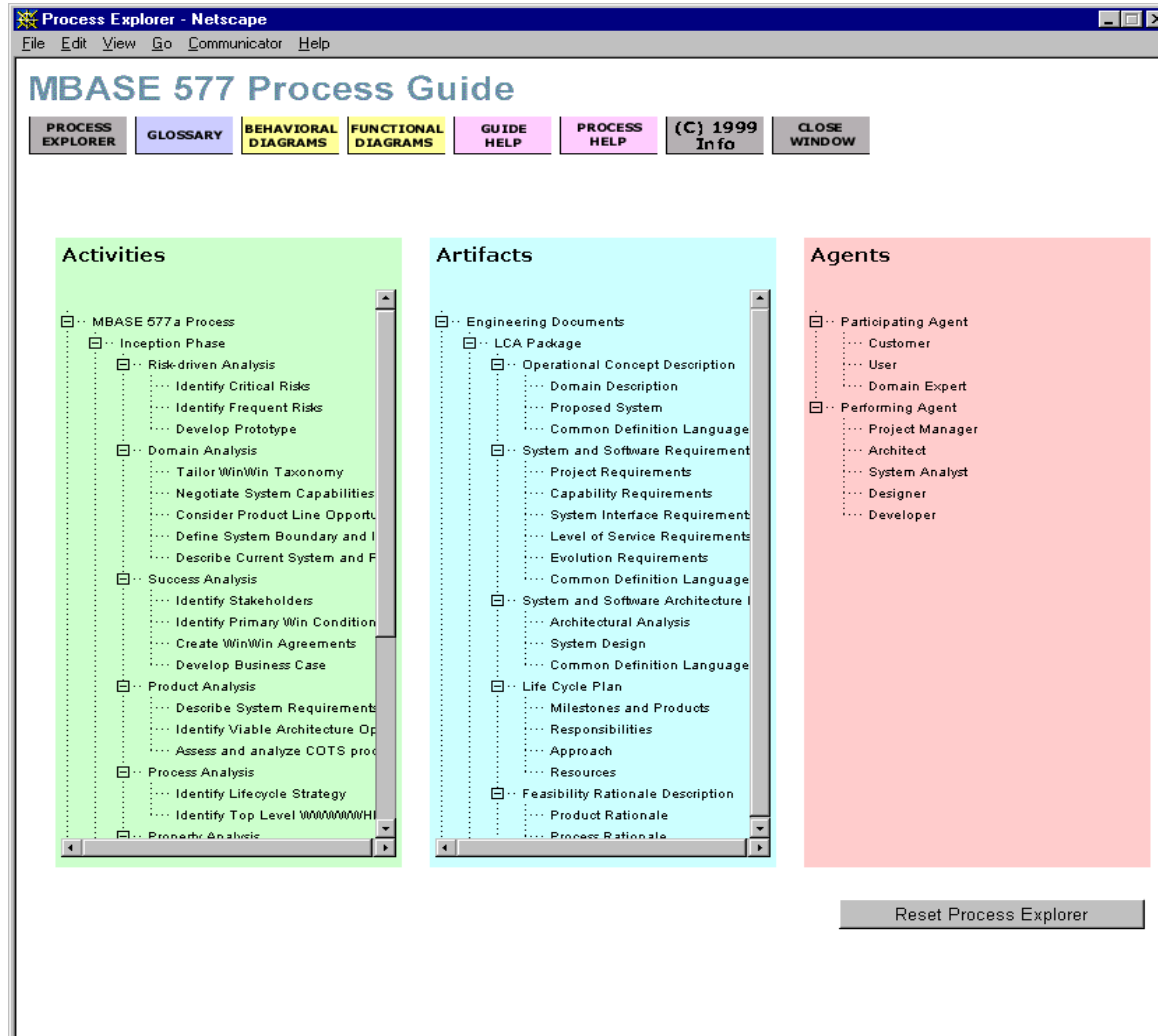
# MBASE Process Framework



# Success Models Drive Other Model Choices

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

# MBASE Electronic Process Guide (1)



The screenshot shows a Netscape browser window titled "Process Explorer - Netscape". The main content area displays the "MBASE 577 Process Guide" with a navigation menu at the top containing buttons for "PROCESS EXPLORER", "GLOSSARY", "BEHAVIORAL DIAGRAMS", "FUNCTIONAL DIAGRAMS", "GUIDE HELP", "PROCESS HELP", "(C) 1999 Info", and "CLOSE WINDOW".

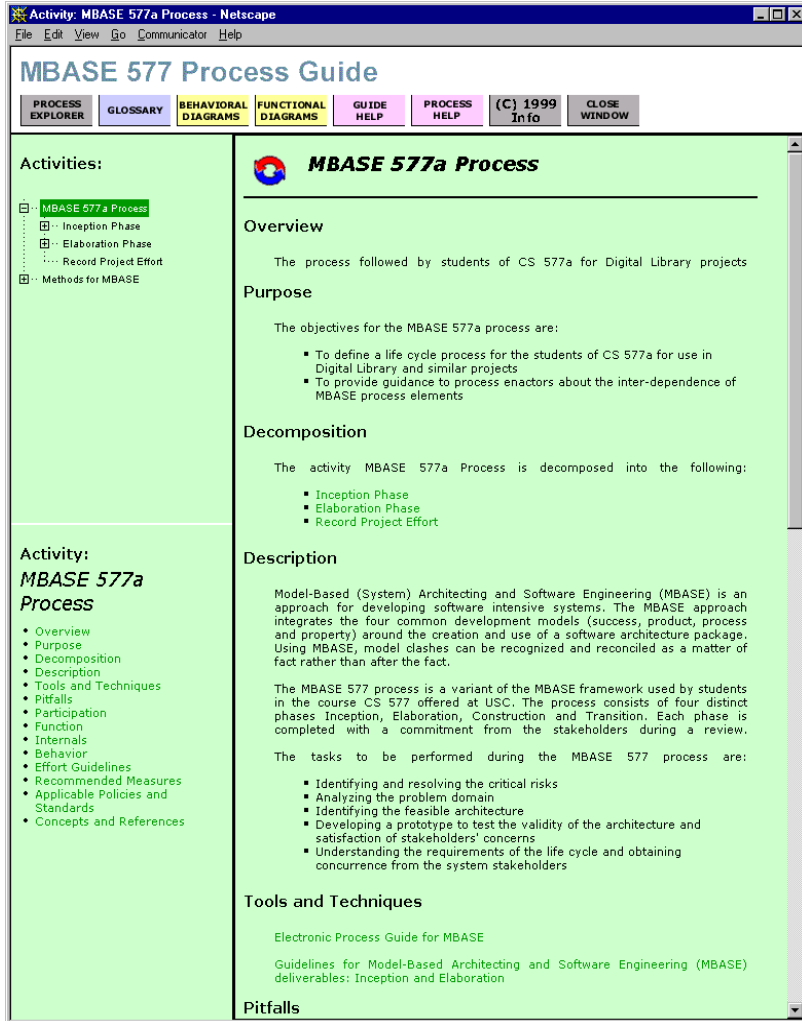
The main content is divided into three vertical panels:

- Activities (Green Panel):**
  - MBASE 577a Process
    - Inception Phase
      - Risk-driven Analysis
        - Identify Critical Risks
        - Identify Frequent Risks
        - Develop Prototype
      - Domain Analysis
        - Tailor WinWin Taxonomy
        - Negotiate System Capabilities
        - Consider Product Line Opport.
        - Define System Boundary and I
        - Describe Current System and F
      - Success Analysis
        - Identify Stakeholders
        - Identify Primary Win Condition
        - Create WinWin Agreements
        - Develop Business Case
      - Product Analysis
        - Describe System Requirements
        - Identify Viable Architecture Op
        - Assess and analyze COTS pro
      - Process Analysis
        - Identify Lifecycle Strategy
        - Identify Top Level WinWin/WHI
      - Property Analysis

- Artifacts (Cyan Panel):**
- Engineering Documents
  - LCA Package
    - Operational Concept Description
      - Domain Description
      - Proposed System
      - Common Definition Language
    - System and Software Requirement
      - Project Requirements
      - Capability Requirements
      - System Interface Requirements
      - Level of Service Requirements
      - Evolution Requirements
      - Common Definition Language
    - System and Software Architecture I
      - Architectural Analysis
      - System Design
      - Common Definition Language
    - Life Cycle Plan
      - Milestones and Products
      - Responsibilities
      - Approach
      - Resources
    - Feasibility Rationale Description
      - Product Rationale
      - Process Rationale
- Agents (Pink Panel):**
- Participating Agent
  - Customer
  - User
  - Domain Expert
- Performing Agent
  - Project Manager
  - Architect
  - System Analyst
  - Designer
  - Developer

At the bottom right of the main content area, there is a button labeled "Reset Process Explorer".

# MBASE Electronic Process Guide (2)



**Activity: MBASE 577a Process - Netscape**

File Edit View Go Communicator Help

## MBASE 577 Process Guide

PROCESS EXPLORER GLOSSARY BEHAVIORAL DIAGRAMS FUNCTIONAL DIAGRAMS GUIDE HELP PROCESS HELP (C) 1999 Info CLOSE WINDOW

**Activities:**

- MBASE 577a Process
  - Inception Phase
  - Elaboration Phase
  - Record Project Effort
  - Methods for MBASE

**Activity: MBASE 577a Process**

- Overview
- Purpose
- Decomposition
- Description
- Tools and Techniques
- Pitfalls

**MBASE 577a Process**

**Overview**

The process followed by students of CS 577a for Digital Library projects

**Purpose**

The objectives for the MBASE 577a process are:

- To define a life cycle process for the students of CS 577a for use in Digital Library and similar projects
- To provide guidance to process enactors about the inter-dependence of MBASE process elements

**Decomposition**

The activity MBASE 577a Process is decomposed into the following:

- Inception Phase
- Elaboration Phase
- Record Project Effort

**Description**

Model-Based (System) Architecting and Software Engineering (MBASE) is an approach for developing software intensive systems. The MBASE approach integrates the four common development models (success, product, process and property) around the creation and use of a software architecture package. Using MBASE, model clashes can be recognized and reconciled as a matter of fact rather than after the fact.

The MBASE 577 process is a variant of the MBASE framework used by students in the course CS 577 offered at USC. The process consists of four distinct phases Inception, Elaboration, Construction and Transition. Each phase is completed with a commitment from the stakeholders during a review.

The tasks to be performed during the MBASE 577 process are:

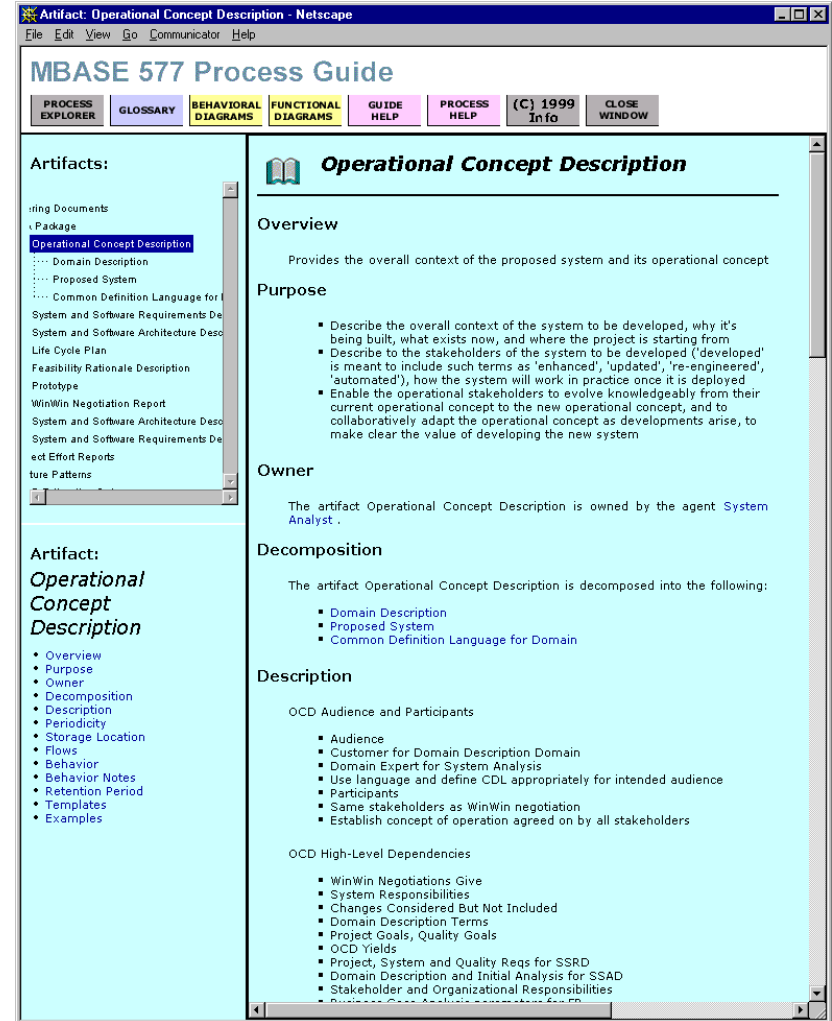
- Identifying and resolving the critical risks
- Analyzing the problem domain
- Identifying the feasible architecture
- Developing a prototype to test the validity of the architecture and satisfaction of stakeholders' concerns
- Understanding the requirements of the life cycle and obtaining concurrence from the system stakeholders

**Tools and Techniques**

Electronic Process Guide for MBASE

Guidelines for Model-Based Architecting and Software Engineering (MBASE) deliverables: Inception and Elaboration

**Pitfalls**



**Artifact: Operational Concept Description - Netscape**

File Edit View Go Communicator Help

## MBASE 577 Process Guide

PROCESS EXPLORER GLOSSARY BEHAVIORAL DIAGRAMS FUNCTIONAL DIAGRAMS GUIDE HELP PROCESS HELP (C) 1999 Info CLOSE WINDOW

**Artifacts:**

- Operating Documents
- Package
  - Operational Concept Description
    - Domain Description
    - Proposed System
    - Common Definition Language for System and Software Requirements Description and Software Architecture Description
    - Life Cycle Plan
    - Feasibility Rationale Description
    - Prototype
    - WinWin Negotiation Report
    - System and Software Architecture Description
    - System and Software Requirements Description
    - Effort Reports
    - Structure Patterns

**Artifact: Operational Concept Description**

- Overview
- Purpose
- Owner
- Decomposition
- Description

**Operational Concept Description**

**Overview**

Provides the overall context of the proposed system and its operational concept

**Purpose**

- Describe the overall context of the system to be developed, why it's being built, what exists now, and where the project is starting from
- Describe to the stakeholders of the system to be developed ('developed' is meant to include such terms as 'enhanced', 'updated', 're-engineered', 'automated'), how the system will work in practice once it is deployed
- Enable the operational stakeholders to evolve knowledgeably from their current operational concept to the new operational concept, and to collaboratively adapt the operational concept as developments arise, to make clear the value of developing the new system

**Owner**

The artifact Operational Concept Description is owned by the agent System Analyst.

**Decomposition**

The artifact Operational Concept Description is decomposed into the following:

- Domain Description
- Proposed System
- Common Definition Language for Domain

**Description**

**OCD Audience and Participants**

- Audience
  - Customer for Domain Description Domain
  - Domain Expert for System Analysis
  - Use language and define CDL appropriately for intended audience
- Participants
  - Same stakeholders as WinWin negotiation
  - Establish concept of operation agreed on by all stakeholders

**OCD High-Level Dependencies**

- WinWin Negotiations Give
  - System Responsibilities
  - Changes Considered But Not Included
  - Domain Description Terms
  - Project Goals, Quality Goals
  - OCD Yields
    - Project, System and Quality Reqs for SSRD
    - Domain Description and Initial Analysis for SSAD
    - Stakeholder and Organizational Responsibilities
    - Business Case Analysis and Justification for MBASE

# The SAIV Process Model

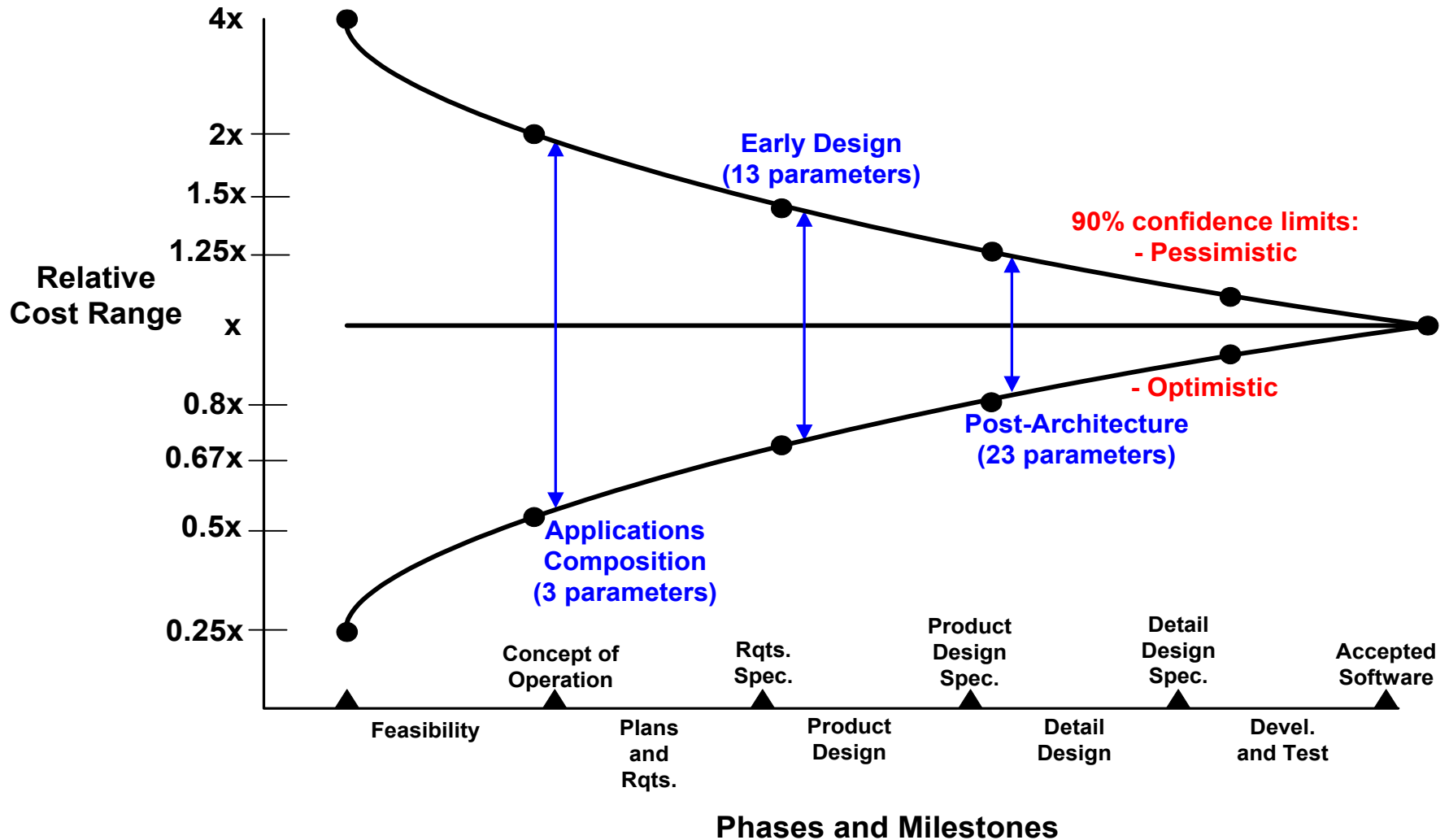
- 1. Shared vision and expectations management**
- 2. Feature prioritization**
- 3. Schedule range estimation**
- 4. Architecture and core capabilities determination**
- 5. Incremental development**
- 6. Change and progress monitoring and control**



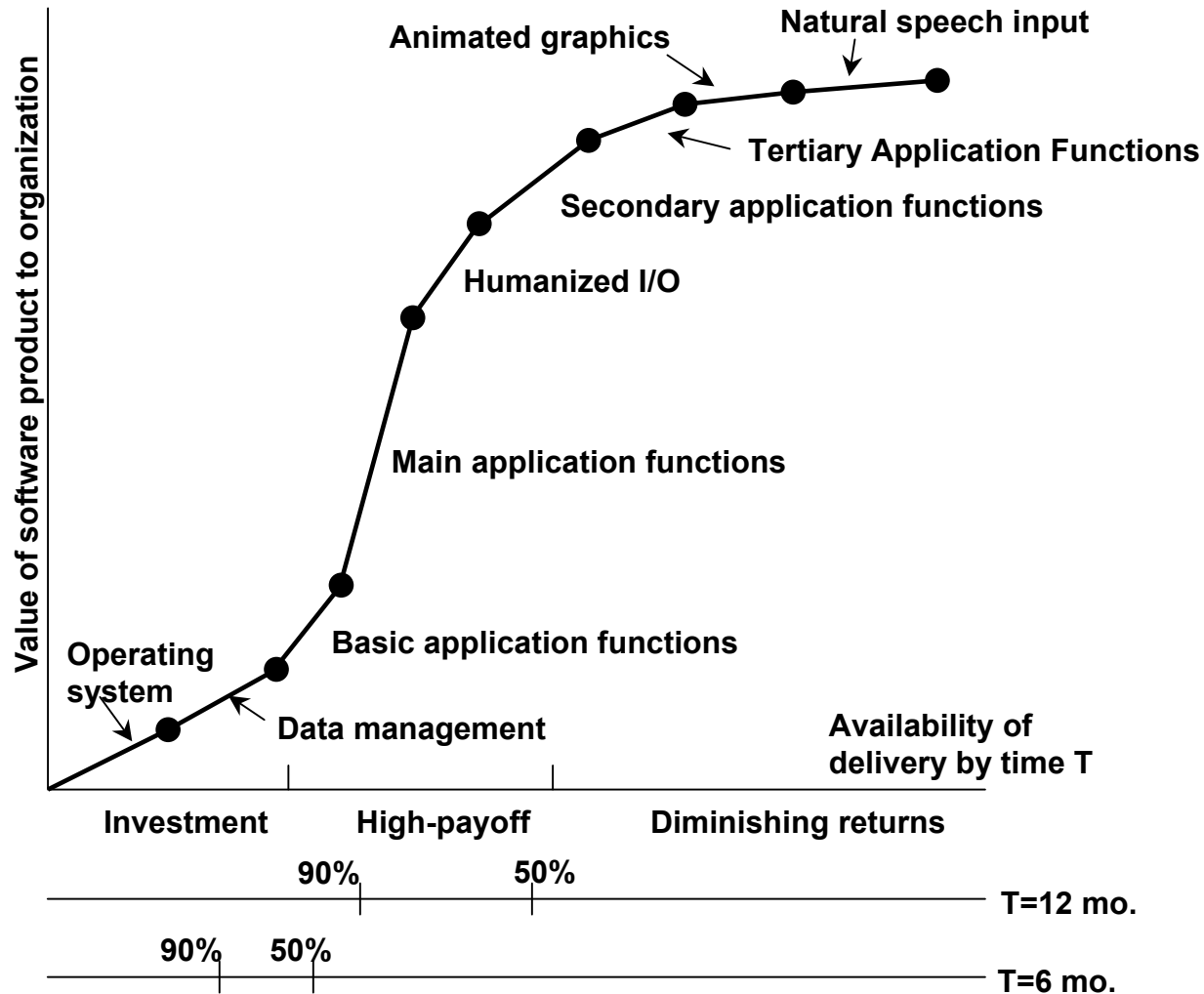
# Shared Vision, Expectations Management, and Feature Prioritization

- **Use stakeholder win-win approach**
- **Developer win condition: Don't overrun fixed 9-month schedule**
- **Clients' win conditions: 24 months' worth of features**
- **Win-Win negotiation**
  - **Which features are most critical?**
  - **COCOMO II: How many features can be built within a 9-month schedule?**

# COCOMO II Estimate Ranges



# Software Product Production Function



# Core Capability Incremental Development, and Coping with Rapid Change

- **Core capability not just top-priority features**
  - Useful end-to-end capability
  - Architected for ease of adding, dropping marginal features
- **Worst case: Deliver core capability in 9 months, with some extra effort**
- **Most likely case: Finish core capability in 6-7 months**
  - Add next-priority features
- **Cope with change by monitoring progress**
  - Renegotiate plans as appropriate



# SAIV Experience I: USC Digital Library Projects

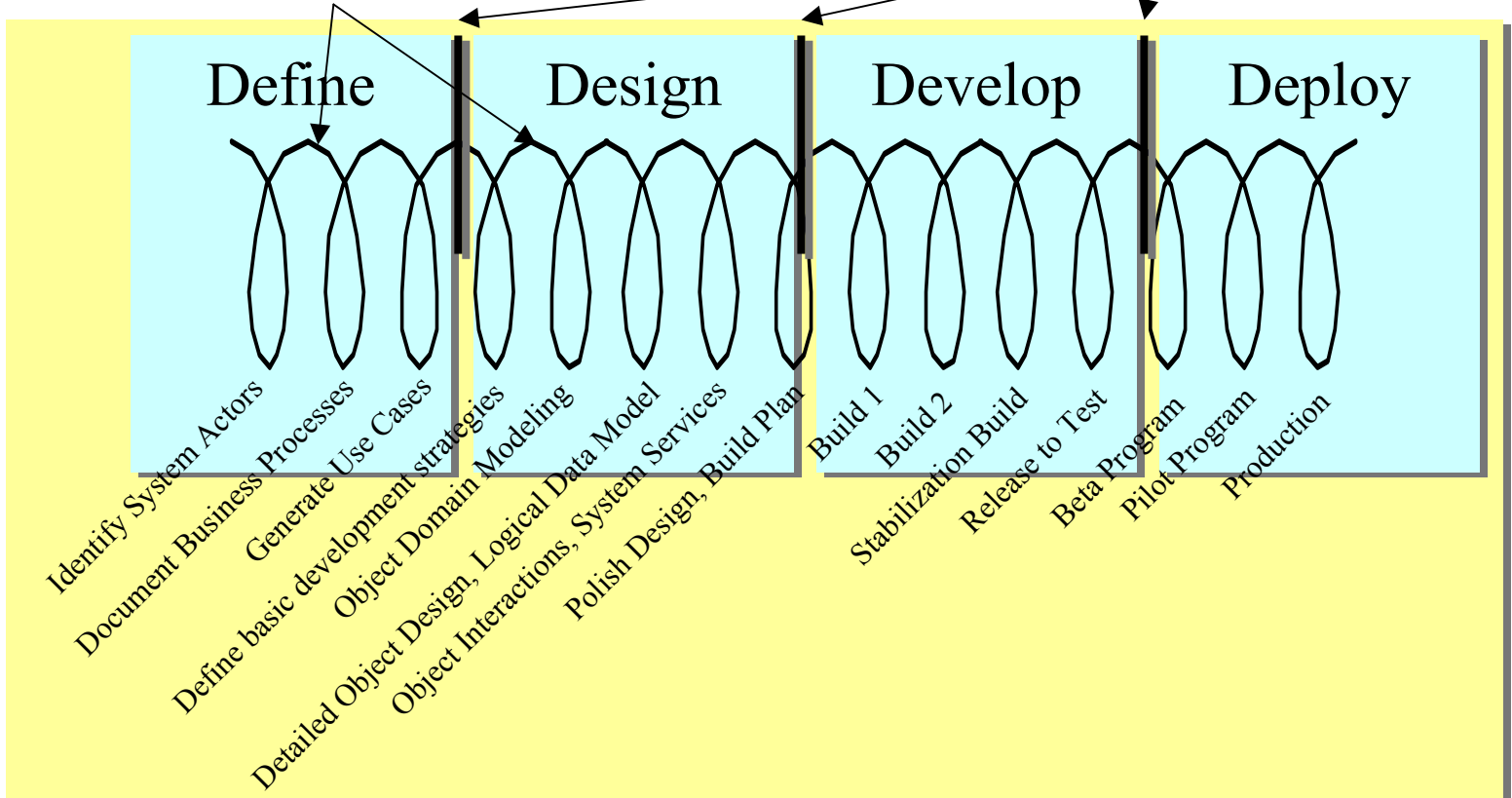
- **Life Cycle Architecture package in fixed 12 weeks**
  - Compatible operational concept, prototypes, requirements, architecture, plans, feasibility rationale
- **Initial Operational Capability in 12 weeks**
  - Including 2-week cold-turkey transition
- **Successful on 17 of 19 projects**
  - Failure 1: too-ambitious core capability
    - Cover 3 image repositories at once
  - Failure 2: team disbanded
    - Graduation, summer job pressures

# Rapid Value™ Project Approach

• 16-24 week fixed schedule

Iteration  
Scope, Listening, Delivery focus

Lines of readiness  
Are we ready for the next step?



# Conclusions: SAIV Critical Success Factors

- **Working with stakeholders in advance to achieve a shared product vision and realistic expectations;**
- **Getting clients to develop and maintain prioritized requirements;**
- **Scoping the core capability to fit within the high-payoff segment of the application's production function for the given schedule;**
- **Architecting the system for ease of adding and dropping features;**
- **Disciplined progress monitoring and corrective action to counter schedule threats**
- **Also works for Cost as Independent Variable**
  - **And “Cost, Schedule, Quality: Pick All Three”**