Spiral Development: Experience, Principles, and Refinements

Barry Boehm, USC
TRW - BMDO Meetings
February 16-17, 2000

boehm@sunset.usc.edu
http://sunset.usc.edu/MBASE
Spiral Model and MBASE

- Spiral experience
  - Critical success factors
    - Invariants and variants
  - Stud poker analogy
- Spiral refinements
  - WinWin spiral
  - Life cycle anchor points
  - MBASE
## Spiral Invariants and Variants - 1

### Critical success factor examples

<table>
<thead>
<tr>
<th>Invariants</th>
<th>Why Invariant</th>
<th>Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Concurrent rather than sequential determination of artifacts (OCD, Rqts, Design, Code, Plans) in each spiral cycle.</td>
<td>• Avoids premature sequential commitments to Rqts, Design, COTS, combination of cost/schedule performance - 1 sec. response time</td>
<td>1a. Relative amount of each artifact developed in each cycle.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1b. Number of concurrent mini-cycles in each cycle.</td>
</tr>
<tr>
<td>2. Consideration in each cycle of critical-stakeholder objectives and constraints, product and process alternatives, risk identification and resolution, stakeholder review and commitment to proceed.</td>
<td>• Avoids commitment to stakeholder-unacceptable or overly risky alternatives.</td>
<td>2a. Choice of risk resolution techniques: prototyping, simulation, modeling, benchmarking, reference checking, etc.</td>
</tr>
<tr>
<td></td>
<td>• Avoids wasted effort in elaborating unsatisfactory alternatives. - Mac-based COTS</td>
<td>2b. Level of effort on each activity within each cycle.</td>
</tr>
<tr>
<td>3. Level of effort on each activity driven by risk considerations.</td>
<td>• Determines “how much is enough” of each activity: domain engr., prototyping, testing, CM, etc. - Pre-ship testing • Avoids overkill or belated risk resolution.</td>
<td>3a. Choice of methods used to pursue activities: MBASE/WinWin, Rational USDP, JAD, QFD, ESP, ...</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3b. Degree of detail of artifacts produced in each cycle.</td>
</tr>
</tbody>
</table>

©USC-CSE
Spiral Invariant I: Concurrent Determination of Key Artifacts (Ops Concept, Rqts, Design, Code, Plans)

• Why invariant
  – Avoids premature sequential commitments to Rqts, Design, COTS, combination of cost/ schedule/ performance
    – 1 sec response time

• Variants
  1a. Relative amount of each artifact developed in each cycle.
  1b. Number of concurrent mini-cycles in each cycle.

• Models excluded
  – Incremental sequential waterfalls with high risk of violating waterfall model assumptions
Sequential Engineering Buries Risk

Arch. A: Custom many cache processors
Arch. B: Modified Client-Server

Original Spec
After Prototyping

$100M
$50M

Response Time (sec)

2/16/00
©USC-CSE
Waterfall Model Assumptions

1. The requirements are knowable in advance of implementation.
2. The requirements have no unresolved, high-risk implications
   - e.g., risks due to COTS choices, cost, schedule, performance, safety, security, user interfaces, organizational impacts
3. The nature of the requirements will not change very much
   - During development; during evolution
4. The requirements are compatible with all the key system stakeholders' expectations
   - e.g., users, customer, developers, maintainers, investors
5. The right architecture for implementing the requirements is well understood.
6. There is enough calendar time to proceed sequentially.
Spiral Invariant 2: Each Cycle Does
objectives, constraints, alternatives, risks,
review, commitment to proceed

• Why invariant
  – Avoids commitment to stakeholder-unacceptable or overly
    risky alternatives.
  – Avoids wasted effort in elaborating unsatisfactory alternatives.
    – Mac-based COTS

• Variants
  2a. Choice of risk resolution techniques: prototyping, simulation,
      modeling, benchmarking, reference checking, etc.
  2b. Level of effort on each activity within each cycle.

• Models excluded
  – Sequential phases with key stakeholders excluded
Mac-Based COTS Example: Digital Library Artifact Viewer

- Great prototype using ER Mapper
  - Tremendous resolution
  - Incremental-resolution artifact display
  - Powerful zoom and navigation features

- Only runs on Unix, Windows
  - Influential Mac user community could sink project
Models Excluded: Sequential Phases Without Key Stakeholders

- High risk of win-lose even with spiral phases
  - Win-lose evolves into lose-lose
- Key criteria for IPT members (AFI 63-123)
  - Representative, empowered, knowledgeable, collaborative, committed
Spiral Invariant 3: Level of Effort Driven by Risk Considerations

• Why invariant
  – Determines ‘how much is enough’ of each activity: domain engr., prototyping, testing, CM, etc.
    – Pre-ship testing
    – Avoids overkill or belated risk resolution.

• Variants
  3a. Choice of methods used to pursue activities: MBASE/WinWin, Rational RUP, JAD, QFD, ESP, . . .
  3b. Degree of detail of artifacts produced in each cycle.

• Models excluded
  – Risk-insensitive evolutionary or incremental development
### Invariants

<table>
<thead>
<tr>
<th>Invariant</th>
<th>Why Invariant</th>
<th>Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Degree of detail of artifacts produced in each cycle driven by risk</td>
<td>- Determines “how much is enough” of each artifact (OCD, Rqts, Design, Code,</td>
<td>4a. Choice of artifact representations (SA/SD, UML, MBASE, formal specs,</td>
</tr>
<tr>
<td>considerations.</td>
<td>Plans) in each cycle.</td>
<td>programming languages, etc.)</td>
</tr>
<tr>
<td></td>
<td>- Avoids overkill or belated risk resolution</td>
<td></td>
</tr>
<tr>
<td>5. Managing stakeholder life-cycle commitments via the LCO, LCA, and IOC</td>
<td>- Avoids analysis paralysis, unrealistic expectations, requirements creep,</td>
<td>5a. Number of spiral cycles or increments between anchor points.</td>
</tr>
<tr>
<td>Anchor Point milestones (getting engaged, getting married, having your</td>
<td>architectural drift, COTS shortfalls and incompatibilities, unsustainable</td>
<td>5b. Situation-specific merging of anchor point milestones.</td>
</tr>
<tr>
<td>first child),</td>
<td>architectures, traumatic cutovers, useless systems.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. Emphasis on system and life cycle activities and artifacts rather than</td>
<td>- Avoids premature suboptimization on hardware, software, or development</td>
<td>6a. Relative amount of hardware and software determined in each cycle.</td>
</tr>
<tr>
<td>software and development activities and artifacts.</td>
<td>considerations.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6b. Relative amount of capability in each life cycle increment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6c. Degree of productization (alpha, beta, shrink-wrap, etc.) of each life</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cycle increment.</td>
</tr>
</tbody>
</table>

©USC-CSE
Spiral Invariant 4: Degree of Detail Driven by Risk Considerations

- **Why invariant**
  - Determines “how much is enough” of each artifact (OCD, Rqts, Design, Code, Plans) in each cycle.
    - Screen layout rqts.
  - Avoids overkill or belated risk resolution.
    - Automated identification of friend or foe (IFF)

- **Variants**
  - 4a. Choice of artifact representations (SA/SD, UML, MBASE, formal specs, programming languages, etc.)

- **Models excluded**
  - Complete, consistent, traceable, testable requirements specification
    - E.g., for GUI, COTS, deferred decisions
Risk-Driven Specifications

- If it's risky not to specify precisely, Do
  - Hardware-software interface
  - Prime-subcontractor interface

- If it's risky to specify precisely, Don't
  - GUI layout
  - COTS behavior
Spiral Invariant 5:
Use of LCO, LCA, IOC, Anchor Point Milestones

• Why invariant
  – Avoids analysis paralysis, unrealistic expectations, requirements creep, architectural drift, COTS shortfalls and incompatibilities, unsustainable architectures, traumatic cutovers, useless systems.

• Variants
  5a. Number of spiral cycles or increments between anchor points.
  5b. Situation-specific merging of anchor point milestones
  • Can merge LCO and LCA when adopting an architecture from 4GL, COTS, product line

• Models excluded
  – Evolutionary development with no life cycle architecture
Life Cycle Anchor Points

• Common System/Software stakeholder commitment points
  – Defined in concert with Government, industry affiliates
  – Coordinated with Rational’s Unified Software Development Process

• Life Cycle Objectives (LCO)
  – Stakeholders’ commitment to support architecting
  – Like getting engaged

• Life Cycle Architecture (LCA)
  – Stakeholders’ commitment to support full life cycle
  – Like getting married

• Initial Operational Capability (IOC)
  – Stakeholders’ commitment to support operations
  – Like having first child
## Win Win Spiral Anchor Points

(Risk-driven level of detail for each element)

<table>
<thead>
<tr>
<th>Milestone Element</th>
<th>Life Cycle Objectives (LCO)</th>
<th>Life Cycle Architecture (LCA)</th>
</tr>
</thead>
</table>
| Definition of Operational Concept | • Top-level system objectives and scope  
- System boundary  
- Environment parameters and assumptions  
- Evolution parameters  
- Operational concept  
- Operations and maintenance scenarios and parameters  
- Organizational life-cycle responsibilities (stakeholders) | • Elaboration of system objectives and scope of increment  
- Elaboration of operational concept by increment |
| System Prototype(s) | • Exercise key usage scenarios  
- Resolve critical risks | • Exercise range of usage scenarios  
- Resolve major outstanding risks |
| Definition of System Requirements | • Top-level functions, interfaces, quality attribute levels, including:  
- Growth vectors and priorities  
- Prototypes  
- Stakeholders' concurrence on essentials | • Elaboration of functions, interfaces, quality attributes, and prototypes by increment  
- Identification of TBD's (to-be-determined items)  
- Stakeholders' concurrence on their priority concerns |
| Definition of System and Software Architecture | • Top-level definition of at least one feasible architecture  
- Physical and logical elements and relationships  
- Choices of COTS and reusable software elements  
- Identification of infeasible architecture options | • Choice of architecture and elaboration by increment  
- Physical and logical components, connectors, configurations, constraints  
- COTS, reuse choices  
- Domain-architecture and architectural style choices  
- Architecture evolution parameters |
| Definition of Life-Cycle Plan | • Identification of life-cycle stakeholders  
- Users, customers, developers, maintainers, interactors, general public, others  
- Identification of life-cycle process model  
- Top-level stages, increments  
- Top-level WWWWHH* by stage | • Elaboration of WWWWHH* for Initial Operational Capability (IOC)  
- Partial elaboration, identification of key TBD's for later increments |
| Feasibility Rationale | • Assurance of consistency among elements above  
- via analysis, measurement, prototyping, simulation  
- Business case analysis for requirements, feasible architectures | • Assurance of consistency among elements above  
- All major risks resolved or covered by risk management |


2/16/00 ©USC-CSE
Initial Operational Capability (IOC)

- **Software preparation**
  - Operational and support software
  - Data preparation, COTS licenses
  - Operational readiness testing
- **Site preparation**
  - Facilities, equipment, supplies, vendor support
- **User, operator, and maintainer preparation**
  - Selection, teambuilding, training
Anchor Points and Rational USDP Phases

<table>
<thead>
<tr>
<th>Engineering Stage</th>
<th>Manufacturing Stage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Feasibility Iterations</strong></td>
<td><strong>LCO</strong></td>
</tr>
<tr>
<td><strong>Architecture Iterations</strong></td>
<td><strong>LCA</strong></td>
</tr>
<tr>
<td><strong>Usable Iterations</strong></td>
<td><strong>IOC</strong></td>
</tr>
<tr>
<td><strong>Product Releases</strong></td>
<td><strong>Management</strong></td>
</tr>
</tbody>
</table>

2/16/00
Evolutionary Development Assumptions

1. The initial release is sufficiently satisfactory to key system stakeholders that they will continue to participate in its evolution.

2. The architecture of the initial release is scalable to accommodate the full set of system life cycle requirements (e.g., performance, safety, security, distribution, localization).

3. The operational user organizations are sufficiently flexible to adapt to the pace of system evolution

4. The dimensions of system evolution are compatible with the dimensions of evolving-out the legacy systems it is replacing.
Spiral Invariant 6:
Emphasis on System and Life Cycle Activities and Artifacts

- Why invariant
  - Avoids premature suboptimization on hardware, software, or development considerations.
  - Scientific American

- Variants
  6a. Relative amount of hardware and software determined in each cycle.
  6b. Relative amount of capability in each life cycle increment
  6c. Degree of productization (alpha, beta, shrink-wrap, etc.) of each life cycle increment.

- Models excluded
  - Purely logical object-oriented methods
    - Insensitive to operational, performance, cost risks
Problems With Programming-Oriented Top-Down Development

"SCIENTIFIC AMERICAN" SUBSCRIPTION PROCESSING

Results:

- More labor-intensive
- Convoluting input controls
- Poor exception-handling
- More errors
- More trivial errors

- More trivial errors
- Greater delays
- Poor exception-handling
- Some input controls
- More labor-intensive

"SCIENTIFIC AMERICAN" SUBSCRIPTION PROCESSING

Center for Software Engineering
University of Southern California
DMR/BRA Results Chain
(DMR Consulting Group’s Benefits Realization Approach)

Order to delivery time is an important buying criterion

ASSUMPTION

INITIATIVE

Implement a new order entry system

Contribution

OUTCOME

Reduced order processing cycle (intermediate outcome)

Contribution

OUTCOME

Reduce time to process order

Increased sales

Reduce time to deliver product

2/16/00
Summary: Hazardous Spiral Look-Alikes

- Incremental sequential waterfalls with significant COTS, user interface, or technology risks
- Sequential spiral phases with key stakeholders excluded from phases
- Risk-insensitive evolutionary or incremental development
- Evolutionary development with no life-cycle architecture
- Insistence on complete specs for COTS, user interface, or deferred-decision situations
- Purely logical object-oriented methods with operational, performance, or cost risks
- Impeccable spiral plan with no commitment to managing risks
Spiral Model and MBASE

- Spiral experience
  - Critical success factors
    - Invariants and variants
  - Stud poker analogy
- Spiral refinements
  - WinWin spiral
  - Life cycle anchor points
- MBASE
Spiral Model and Incremental Commitment: Stud Poker Analogy

- Evaluate alternative courses of action
  - Fold: save resources for other deals
  - Ante: buy at least one more round

- Using incomplete information
  - Hole cards: competitive situation
  - Rest of deck: chance of getting winner

- Anticipating future possibilities
  - Likelihood that next round will clarify outcome

- Commit incrementally rather than all at once
  - Challenge: DoD POM process makes this hard to do
• Where do objectives, constraints, alternatives come from?
  - Win Win extensions
• Lack of intermediate milestones
  - Anchor Points: LCO, LCA, IOC
  - Concurrent-engineering spirals between anchor points
• Need to avoid model clashes, provide more specific guidance
  - MBASE
Success Model-Clash Profiles: General

**Users**
- Many features
- Changeable requirements
- Applications compatibility & control
- High levels of Service
- Voice in acquisition
- Flexible contract

**Maintainers**
- Ease of transition
- Ease of maintenance
- Applications compatibility & control
- Voice in acquisition

**Acquirers**
- Mission cost/effectiveness
- Limited budget, Schedule
- Government standards compliance
- Political correctness
- Development visibility & control
- Rigorous contract

**Developers**
- Flexible contract
- Ease of meeting budget & schedule
- Stable requirements
- Freedom of choice: process
- Freedom of choice: team
- Freedom of choice: COTS/reuse

©USC-CSE
MBASE Process Framework

- Stakeholders
  - Determine the relevance of
  - Identify, prioritize
- Success Models
  - Enable satisficing among
- Domain/Environment Models
- Property Models
  - Provide evaluations for
- Conceptual Product Models
  - Are refinements of
  - Intermediate Product Models
  - Reified Product Models
- WinWin Spiral Process
- Life Cycle
  - Architecture Package
  - Plan in LCA Package
- Impose constraints
-进步在选择和思考
- Serve and satisfy

2/16/00 ©USC-CSE
# Success Models Drive Other Model Choices

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

OUSC-CSE

Domain constrained by schedule; architected for ease in dropping features to meet schedule

©USC-CSE

2/16/00
Activities

- MBASE 577 Process Activities
  - Inception Phase:
    - Risk-driven Analysis
    - Identify Customer Requirements
    - Identify Project Milestones
    - Develop Preliminary Requirements
    - Domain Analysis
    - Task/Work Taxonomy
    - Identify System Capabilities
    - Identify Current System and P
    - Identify Stakeholders
    - Identify Primary Win Conditions
    - Create Win-Win Agreement
    - Identify Failure Scenarios
    - Describe Current System and P
  - Product Analysis:
    - Describe Product Requirements
    - Identify Available Alternatives
    - Name and define MBASE 577
    - Develop MBASE 577
    - Identify Change Strategy
    - Identify Key Issues

Artifacts

- Engineering Documents:
  - Operational Concept Description
  - System Description
  - System Interface Requirement
  - Proposed System Design
  - Common Definition Language

- System and Software Attributes:
  - Antecedent Analysis
  - System Design
  - Common Definition Language

- System Development:
  - MBASE Plan
  - Milestones and Products
  - Responsibilities
  - Approach
  - Resources
  - Feasibility Rationale
  - Product Rationale

Agents

- Participating Agent:
  - Customer
  - User
  - Domain Expert

- Performing Agent:
  - Project Manager
  - Systems Analyst
  - Designer
  - Developer
MBASE Electronic Process Guide (2)

Overview

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

Purpose

- To define the work process for the students of CS 577a for use in Digital Library projects and similar projects
- To guide project leaders towards understanding about the inter-dependence of process elements

Decomposition

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

- Model-Based System Architecture and Software Engineering (MBASE)
- Process Guide (2)
- Artifacts:
  - Operational Concept Description

Description

MBASE 577a Process

- Purpose: To define the work process for the students of CS 577a for use in Digital Library projects and similar projects
- Decomposition: The activity MBASE 577a Process is decomposed into the following:
  - Model-Based System Architecture and Software Engineering (MBASE)
  - Process Guide (2)
  - Artifacts:
    - Operational Concept Description

Operational Concept Description

- Overview: Provides the overall context of the proposed system and its operational concept
- Purpose: To define the overall context of the system to be developed, how it will be developed, and how it will be used
- Decomposition: The overall Operational Concept Description is decomposed into the following:
  - Overview
  - Process Guide (2)
  - Artifacts:
    - Operational Concept Description

2/16/00 ©USC-CSE
## MBASE Project Experience at USC/Columbia

### Fall Semester: LCA Package
- **Teams:** 15
- **Students:** 86
- **Applications:** 12
- **Teams failing LCO review:** 4
- **Pages, LCO package:** 160
- **Client Evaluation (1-5, 5 best):** 4.46

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Teams</td>
<td>15</td>
<td>16</td>
<td>20</td>
<td>20</td>
<td>13</td>
</tr>
<tr>
<td>Students</td>
<td>86</td>
<td>80</td>
<td>102</td>
<td>107</td>
<td>59</td>
</tr>
<tr>
<td>Applications</td>
<td>12</td>
<td>15</td>
<td>17</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Teams failing LCO review</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>Pages, LCO package</td>
<td>160</td>
<td>103</td>
<td>114</td>
<td>124</td>
<td>116</td>
</tr>
<tr>
<td>Client Evaluation (1-5, 5 best)</td>
<td>4.46</td>
<td>4.67</td>
<td>4.74</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

### Spring Semester: IOC Package
- **Teams:** 6
- **Students:** 28
- **Applications:** 8
- **Teams failing IOC acceptance review:** 0
- **Applications satisfying clients:** 5
- **Applications not overtaken by events:** 6
- **Applications continued:** 3
- **Applications used:** 1
- **Client evaluation:** -

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Teams</td>
<td>6</td>
<td>5</td>
<td>6</td>
<td>Remained the same</td>
<td>since projects were only one semester long</td>
</tr>
<tr>
<td>Students</td>
<td>28</td>
<td>23</td>
<td>28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applications</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Teams failing IOC acceptance review</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Applications satisfying clients</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>20*</td>
<td>12*</td>
</tr>
<tr>
<td>Applications not overtaken by events</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>Applications continued</td>
<td>3</td>
<td>3</td>
<td>TBD</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Applications used</td>
<td>1</td>
<td>3</td>
<td>TBD</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>Client evaluation</td>
<td>-</td>
<td>4.15</td>
<td>4.3</td>
<td>4.44</td>
<td>4.21</td>
</tr>
</tbody>
</table>

2/16/00
CCPDS-R MBASE Models

• Success Models
  – Reinterpreted DOD-STD-2167a; users involved
  – Award fee flowdown to performers

• Product Models
  – Domain model and architecture
  – Message-passing middleware (UNAS)

• Process Models
  – Ada process model and toolset
  – Incremental builds; early delivery

• Property Models
  – COCOMO cost & schedule
  – UNAS - based performance modeling
  – Extensive progress and quality metrics tools
Integrating DMR/BRA and MBASE

Stakeholders

- provide value realization
- feedback to

Success Models

- provide measures for
- constrain

Business Initiatives (Process, product)

- contribute to

IT Initiatives (process, product)

- constrain the realization of

Business Assumptions

- constrain the realization of

Business Value Assumptions

- contribute to

Business Value Outcomes

Potential Actual

- (model)
- (performance)

Development Assumptions

©USC-CSE
# MBASE Invariants and Variants

<table>
<thead>
<tr>
<th>Invariants</th>
<th>Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Defining and sustaining a stakeholder win-win relationship through the system's life-cycle.</td>
<td>1. Use of particular success, process, product, or property models.</td>
</tr>
<tr>
<td>2. Using the MBASE Model Integration Framework.</td>
<td>2. Choice of process or product representation.</td>
</tr>
<tr>
<td>4. Using the LCO, LCA, and IOC Anchor Point milestones.</td>
<td>4. Number of spiral cycles or builds between anchor points.</td>
</tr>
<tr>
<td>5. Ensuring that the content of MBASE artifacts and activities is risk-driven.</td>
<td>5. Mapping of activities onto Inception-Elaboration-Construction-Transition phases.</td>
</tr>
<tr>
<td>6. Mapping of staff levels onto activities.</td>
<td></td>
</tr>
</tbody>
</table>
Conclusions: Transitioning to CMMI

- Need a model that is
  - Well-grounded in software experience
  - Supportive of all CMMI process areas
  - Tailorable to individual situations
  - Specific about what to do
  - Supportive of future software/system trends
    - Rapid Application Development (RAD)
    - Cost/Schedule as Independent Variable (CAIV/SAIV)
    - COTS Integration
    - Web, Internet, Agents, et al.

- Spiral Model? Almost. Needs some refinements

- MBASE? Provides key refinements
(MBASE material available at http://sunset.usc.edu/MBASE)


