Model Driven Development Needs More Than Product Models

Barry Boehm, USC

USC-CSE Executive Workshop on MDA
Mar. 16th, 2005
Outline

• Nature of Model Clashes
  • Among product, process, property, success models
  • Project examples

• Distribution of Model Clashes
  • 35-project sample: Al-Said Ph.D. thesis
  • Product-product: 30% of total, 24% of risk

• Detecting and Avoiding Model Clashes
  • MBASE integration and process frameworks
    • Concurrent engineering of architecture, rqt’s, plans
  • Anchor point milestones and Feasibility Rationale

• Conclusions
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
Using Models to Visualize Software Facets

Success Models
- Win-Win
- Business Case Analysis
- Results Chains
- Risk
- Software Warranties
- Correctness
- RAD
- Six Sigma
- Stories
- Award Fees
- Agility
- JAD
- QFD
- Golden Rule

Product Models
- UML
- XML
- CORBA
- COM
- Architectures
- Product Lines
- OO Analysis & Design
- Requirements
- Operational Concepts
- Domain Ontologies
- COTS
- GOTS

Process Models
- Waterfall
- Spiral
- RUP
- XP
- SAIV
- CAIV
- SCQAIV
- Risk Management
- Business Process Reengineering
- CMM’s
- Peopleware
- IPT’s
- Agile Development
- Groupware
- Easy WinWin
- Experience Factory
- GQM

Property Models
- COCOMO II
- COCOTS
- CORADMO
- System Dynamics
- Metrics
- -ilities
- COQUALMO
- Simulation and Modeling

MBASE, CeBASE
Model Clash Example: Waterfall and COTS
- e.g., DoD-STD-2167 and SecDef Perry COTS memo

- Waterfall Process Model Assumptions
  - System requirements are specifiable in advance
  - They determine the system’s capabilities

- COTS-Based Product Model Assumptions
  - COTS capabilities determine affordability
  - Affordability determines requirements
    • It’s not a requirement if you can’t afford it
Large Government Waterfall Project Example

- **Original Cost**
  - $50M
  - $100M

- **Response Time (sec)**
  - Original Spec 1
  - 2
  - 3
  - 4
  - 5

- **Architecture**
  - Original: Modified Client-Server
  - Required: Custom; many cache processors

- **After Prototyping**
# Clashes Among MBASE Models

<table>
<thead>
<tr>
<th>Product Model</th>
<th>Process Model</th>
<th>Property Model</th>
<th>Success Model</th>
</tr>
</thead>
</table>
| **Product Model** | • Structure clash  
• Traceability clash  
• Architecture style clash | • COTS-driven product vs. Waterfall (requirements-driven) process | • 4GL-based product vs. low development cost and performance scalability |
| **Process Model** | • Multi-increment development process vs. single-increment support tools | • Evolutionary development process vs. Rayleigh-curve cost model | • Waterfall process model vs. "I'll know it when I see it" (IWIWISI) prototyping success model |
| **Property Model** | • Minimize cost and schedule vs. maximize quality (Quality is free) | | • Fixed-price contract vs. easy-to-change, volatile requirements |
| **Success Model** | | • Golden Rule vs. stakeholder win-win | |
Model Clashes SpiderWeb: MasterNet

Users
- Many features
- Changeable requirements
- Applications compatibility
- High levels of service
- Voice in acquisition
- Flexible contract
- Early availability

Maintainers
- Ease of transition
- Ease of maintenance
- Applications compatibility
- Voice in acquisition

Acquirers
- Mission cost/effectiveness
- Limited development budget, schedule
- Government standards compliance
- Political correctness
- Development visibility and control
- Rigorous contact

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

PC: Process
PD: Product
PP: Property
S: Success
Outline

- **Nature of Model Clashes**
  - Among product, process, property, success models
  - Project examples

- **Distribution of Model Clashes**
  - 35-project sample: Al-Said Ph.D. thesis
  - Product-product: 30% of total, 24% of risk

- **Detecting and Avoiding Model Clashes**
  - MBASE integration and process frameworks
    - Concurrent engineering of architecture, rqt’s, plans
    - Anchor point milestones and Feasibility Rationale

- **Conclusions**
## Model Clashes Occurrence Probability & Severity: 35 Projects RAD Sample

<table>
<thead>
<tr>
<th>Assumptions</th>
<th>S</th>
<th>M</th>
<th>OP</th>
<th>SV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer needs extensive, on demand user/customer interaction</td>
<td>Dev</td>
<td>SS</td>
<td>0.86</td>
<td>H</td>
</tr>
<tr>
<td>User/Customer are available at limited times</td>
<td>User/Cust</td>
<td>SS</td>
<td></td>
<td>H</td>
</tr>
<tr>
<td>User/Customer are free to add/modify the system requirements during the system implementation</td>
<td>User/Cust</td>
<td>PD</td>
<td>0.46</td>
<td>H</td>
</tr>
<tr>
<td>Changes/additions to system requirements require extra budget and schedule</td>
<td>Dev</td>
<td>PP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S: Stakeholder; M: Model; OP: Occurrence Probability; SV: Severity
Clash Types and their Contribution to Project Risk

- Majority of research (product-product) addresses minority of risk
Inter and Intra Model Clashes and their Contribution to Project Risk

- Inter model clashes caused majority of risk
Model Clash Analysis Tool

Add Question: Type Question and its properties

Model Comparison Results

<table>
<thead>
<tr>
<th>Waterfall Assumption</th>
<th>COTS Assumption</th>
<th>Risk</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer defines and documents a complete set of requirements for each system capability before starting the system detail design</td>
<td>System requirements and architectures are often driven by the capabilities and structures of the final set of COTS components selected for integration</td>
<td>Defining the complete requirements first and then looking for COTS product(s) to satisfy them has many problems. These problems are caused by the inflexibility of COTS. As an example, available COTS product(s) may not satisfy a requirement that demands a specific performance</td>
<td>Before starting the system design, prioritize the system requirements and then assess viable COTS products that satisfy these requirements. II. All key stakeholders should be involved in assessing potential COTS. III. Finalize the system requirements specifications after knowing COTS limitations</td>
</tr>
</tbody>
</table>
Outline

• Nature of Model Clashes
  • Among product, process, property, success models
  • Project examples

• Distribution of Model Clashes
  • 35-project sample: Al-Said Ph.D. thesis
  • Product-product: 30% of total, 24% of risk

Detecting and Avoiding Model Clashes
  • MBASE integration and process frameworks
    • Concurrent engineering of architecture, rqt’s, plans
  • Anchor point milestones and Feasibility Rationale

• Conclusions
MBASE Integration Framework

Process models

- Life cycle anchor points
- Risk management
- Key practices

Success models

- Business case
- IKIWISI
- Stakeholder win-win

- Process entry/exit criteria

Product models

- Product evaluation criteria

Property models

- Planning and control
- Milestone content
- Evaluation and analysis

- Domain model
- Requirements
- Architecture
- Code
- Documentation

Product models

- Cost
- Schedule
- Performance
- Reliability
MBASE Process Framework

- Stakeholders
  - set context for
  - determine the relevance of

- Domain/Environment Models
  - provide parameters for
  - are refinements of Conceptual Product Models

- Conceptual Product Models
  - IPM₁
  - IPM₂
  - Reified Product Models
  - influence selecting, and serve and satisfy
  - reify ...

- Success Models
  - provide evaluations for
  - impose constraints on
  - enable satisficing among

- Property Models
  - provide parameters for

- Process Models
  - guide progress in selecting, and reifying
  - serve and satisfy

- WinWin Spiral Process
- Life Cycle
- Architecture Package
- Plan in LCA Package

- MBASE Process Framework
Spiral Anchor Points Enable Concurrent Engineering

Disciplines
- Business Modeling
- Requirements
- Analysis & Design
- Implementation
- Test
- Deployment
- Configuration & Change Mgmt
- Project Management
- Environment

Phases
- Inception
- Elaboration
- Construction
- Transition

Iterations
- Initial
- Elab #1
- Elab #2
- Const #1
- Const #2
- Const #N
- Tran #1
- Tran #2
### Need Concurrently Engineered Milestone Reviews

**Life Cycle Objectives (LCO); Life Cycle Architecture Package (LCA)**

| Operational Concept | • Elaboration of system objectives and scope by increment  
|                     | • Elaboration of operational concept by increment |
| System Prototype(s) | • Exercise range of usage scenarios  
|                     | • Resolve major outstanding risks |
| System Requirements | • Elaboration of functions, interfaces, quality attributes, and prototypes by increment  
|                     | - Identification of TBD’s (to be determined items) |
|                     | • Stakeholders’ concurrence on their priority concerns |
| System and Software Architecture | • 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 |
Need Concurrently Engineered Milestone Reviews
Life Cycle Objectives (LCO); Life Cycle Architecture Package (LCA)\textsubscript{2}

| Life-Cycle Plan | • Elaboration of WWWWWHH\textsuperscript{*} for Initial Operational Capability (IOC)  
|                 | • Partial elaboration, identification of key TBD’s for later increments |
| Feasibility Rationale | • Assurance of consistency among elements above  
|                     | • All major risks resolved or covered by risk management plan. |

\textsuperscript{*}WWW\textsuperscript{WW}WHHH: Why, What, When, Who, Where, How, How Much
LCO (MS A) and LCA (MS B) Pass/Fail Criteria

- A system built to the given architecture will
  - Support the operational concept
  - Satisfy the requirements
  - Be faithful to the prototype(s)
  - Be buildable within the budgets and schedules in the plan
  - Show a viable business case
  - Establish key stakeholders’ commitment to proceed

LCO: True for at least one architecture
LCA: True for the specific life cycle architecture;
   All major risks resolved or covered by a risk management plan
Conclusions

• Need to address model clashes among product, process, property, and success models
  – A major hidden source of project failures
  – Product-product clashes: 30% of total, 24% of risk

• Best to concurrently engineer operational concept, requirements, architecture, plans, feasibility rationale

• Win-Win spiral model and anchor point milestones enable controlled concurrent engineering
  – Life Cycle Architecture Package vs. standalone architecture
  – 93% success rate of on-time client satisfaction on campus e-services projects