Software Engineering in CREATE – Lessons Deployed

The Application of Engineering Rigor to Software Development—
Systems Engineering for Software

Richard P Kendall, Ph.D. (Presented by Dr. Saikat Dey)
November 2012

Distribution Statement A: Approved for public release; distribution is unlimited.
Gov’t Software: A Legacy of Risk Management Failure!
HPCMP Computers, Networks and CREATE Tools Bridge the Gap over the “Valley of Death”

Death Valley, CA

Budget Authority

1. Basic Research
2. Applied Research
3. Advanced Technology Development

4. Advanced Component Development and Prototype
5. System Development and Demonstration
6. RDT&E Management Support
7. Operational Systems Development

R.I.P. S&T

So-Called “Valley of Death”

HPCMP-CREATE Bridge
Compounding Risk Factors

The Four CREATE Program Complexities:

1. Complex Physics (Integrated Multi-Scale, Multi-Physics)
2. Complex Computing (networks, security, architectures)
3. Complex Development Organizations (Distributed)
4. Complex Customers (Multi-Service, Multi-Community)
Managing Risks in CREATE

Software Engineering provides the framework for managing risk

- Based on experience from DoD, DOE, Industry, Academia case studies
- Adapts best practices to physics-based software:
  - Program Management
  - Requirements Management
  - Configuration Management
  - Quality Assurance
  - Verification, Validation and Uncertainty Quantification

after CMMI (SEI)
An Example Similar to CREATE

- **ASCI (Multi-Physics, HPC) < 50% Success**

CREATE Scale

Risk Factors: Why Software Projects fail.

1. Project complexity
2. Project goals
3. Acquiring needed resources and skills
4. Requirements synthesis
5. Reporting
6. Communication among developers, customers and users
7. Technology maturity
8. Development practices
9. Project management, especially Planning & Execution
10. Stakeholder/Customer politics

CREATE Core Software Engineering Practices

Development Team
1. Lean (<10), close-knit development teams led by technical experts. [3,8,10]

2. Emphasis on transparency in development across CREATE projects. [6]

Customer Focus
3. Stakeholder-driven requirements through Boards of Directors comprised of stakeholder and user representatives. [2,4,6,10]

4. Pilots to solicit customer reaction and input to feature and attribute implementations. [4]

5. Frequent reporting to stakeholders. [6]
CREATE Core Software Engineering Practices

**Technical Maturity**
6. Reliance on proven technologies to satisfy customer-defined use cases. [1,2,7]
7. VVUQ in alignment with NRC recommendations for scientific code. [10]

**Development Methods**
8. Milestone-driven workflow management with flexible workflow execution and annual releases. [8,9]

9. Configuration management, including configuration control boards (CCBs), code management, build automation, continuous integration, and issue tracking. [8]

10. No code checked into the development branch without an accompanying test. [8]

11. Documented code with user’s manuals, technical descriptions, tutorials, example problem setup and user forums. [6]
CREATE Core Software Engineering Practices

Requirements Definition
12. Reliance on prototypes to solidify difficult-to-specify, or possibly ambiguous requirements. [4]
Lean, Distributed, Expert-led Teams

CREATE Project Sites
- AV Ships RF MG
- U of Wy
- U of Iowa
- U of Mich
- ASC
- AFRL
- SPAWAR
- SNL
- AEDC
- ERDC
- 46th Test Wing
- CERDEC
- Pax River
- HPC
- Carderock
- NAVSEA
- Indian Head
- ONR
- NRL

Geographic

Embedded in Stakeholder Orgs

Organizational

HPCMP Director
John West
CREATE Program
Douglass Post

Official HPCMP Advisory Panel

Ships Project
Myles Hurwitz
Carderock

NavyFOAM
Joseph Gorski
Carderock

Integrated Hydro Design Environment
Joseph Gorski
Carderock

Navy Enhanced Sierra Mechanics
Tom Moyer
Carderock

Rapid Ship Design Environment
Adrian McKenna
Carderock

Air Vehicles Project
Robert Meakin
NAVAIR, Patuxent River

Kestrel
Scott Morton
Eglin AFB

Shadow-Ops
Joe Laiosa
NAVAIR, Pax River

Helios
Roger Strawn
Army, Ames

DaVinci
Greg Roth
ASC & AFRL

Firebolt
Robert Nichols
AEDC

RF Antennas Project
John D'Angelo
AFRL, WPAFB

SENTRI-core
AFRL, WPAFB

Mesh & Geometry Project
Saikat Dey
NRL
Capstone

SENTRI-V&V
Andrew Greenwood
AFRL, KAFB

Distribution C. Please see Page 1 for additional information
Transparency

Welcome to CREATE

Please login to access CREATE Projects

Air Vehicles (AV)
- Military air vehicle design and analysis
  - Software: Kestrel, Helios, DaVinci

Meshing and Geometry (MG)
- Capstone Modeling, Geometry, and Meshing
  - Software: Capstone

Portal Applications (PA)
- HPCMP Portal Initiative

RF Antennas (RF)
- RF antenna design and integration with platforms
  - Software: SENTRI

SEC (SH)
- Network security, and encryption

US GOVERNMENT NOTICE AND CONSENT

You are accessing a U.S. Government (USG) Information System (IS) that is
limited to, penetration testing, COMSEC monitoring, network operations,
and countersurveillance (CI).

At any time, the USG may inspect and seize data stored on this IS.

Communications using, or stored on this IS are not private, nor
protected, and may be disclosed or used for any USG interest—neither for your personal
benefit or privacy.

Notwithstanding the above, using this IS does not constitute consent
for searches or monitoring of the content of privileged communications,
personal representation or services by attorneys, psychologists, and
such communications and work product are private and confidential.

The Login button indicates your acceptance of the above.

https://portal.create.hpc.mil
Product Architectures Visible Across CREATE
Customer-Driven Use Cases

- Capture Requirements in customer-oriented language
- Clearly identify the “user”
- Describe the Goal of the user
- Specify Minimum functionality or performance expectations
- Describe main success scenarios

[from Cockburn, 2001, “fully-dressed” use cases]
Customer Driven “Use Cases”

Use Case Example: Ship Shock (NESM)

Support Survivability Analysis (by Navy Structural Engineers) from Shock Damage from Explosives when

• Use Case I
  — Structural Response is Essentially Linear Elastic
  — Local Nonlinearities (mounts, joints, etc.)

• Use Case II (includes SURFEX)
  — Structural Response is Elastic/Plastic w/ Damage
  — Local Nonlinearities Included

• Use Case III
  — Same Characteristics As UC II w/ Addition Of Cascading Structural Damage (debris)
CREATE Project Roadmaps

Example from Ships Hydro

- Resistance Related
  - UCR1: Hull with fixed ship sinkage and trim
  - UCR2: Hull with computed sinkage and trim
- Powering Related
  - UCP1: Body force model for propulsor
  - UCP2: Full propulsor/hull modeling
- Maneuvering Related (motions in calm water)
  - UCM1: Rotating arm steady turning motion
  - UCM2: Planar Motion Mechanism (PMM)
  - UCM3: Moving appendages and controller
- Seakeeping Related (involves waves)
  - UCS1: Prescribed trajectory in regular waves
  - UCS2: Hull responds to regular waves
  - UCS3: Prescribed trajectory in irregular waves
  - UCS4: Predicted motions with moving appendages in waves
  - UCS5: Seaway loads with one way coupling to structures code
  - UCS6: Seaway loads with two way coupling to structures code
CREATE-AV Process to Manage Capability Gaps (Customer Requirements)

1 – Identify Key Acquisition Processes (AP’s)

2 – Identify Products of AP’s

3 – Breakdown AP Workflows (WF’s)

4 – Identify HPC Insertion Points into WF’s

5 – Identify HPC Analysis Capabilities required to improve AP WF’s

6 – Prioritize and Group analysis capabilities

7 – Select Groups that represent greatest impacts to acquisition for HPC software development under CREATE-AV

Approved by BoD

AV Planning Team=Senior Customer Engineers

Updated annually.

Sr. Management +AV Planning Team

8 – Build mechanisms for CREATE-AV software to impact targeted AP’s

Presentation Title
Page-17

Distribution C. Please see Page 1 for additional information
Pilots in AV

Annually execute between 4 and 6 Pilot Projects to “shadow” acquisition programs engineering workflows– 26 Pilots since 2008!

- **Build bridges** of trust between product developers and targeted acquisition engineering orgs in order to deploy CREATE-AV technology
- **Learn workflows** and actual requirements of targeted orgs
- Key roles in product VV&QA (Verification, Validation, & Quality Control)
  - Build computational baselines
  - Build **archive of validation cases** (VERY big deal)
Multi-Physics based on Proven Technologies

HYDRO Design Environment

Spectral Ocean Wave Model

KESTREL + Helios

CFD + Combustion

Structural Dynamics

CFD

Presentation Title

Page-19

Distribution C. Please see Page 1 for additional information
Verification Principles and Best Practices

- Principle: Solution verification is well defined only in terms of specified quantities of interest, which are usually functionals of the full computed solution.
  - Best practice: Clearly define the QOIs for a given VVUQ analysis, including the solution verification task. Different QOIs will be affected differently by numerical errors. CREATE VVUQ Practice 11
- Best practice: Ensure that solution verification encompasses the full range of inputs that will be employed during UQ assessments. CREATE VVUQ Practice 8
- Principle: The efficiency and effectiveness of code and solution verification can often be enhanced by exploiting the hierarchical composition of codes and mathematical models, with verification performed first on the lowest-level building blocks and then successively on more complex levels.
  - Best practice: Identify hierarchies in computational and mathematical models and exploit them for code and solution verification. It is often worthwhile to design the code with this approach in mind. CREATE VVUQ Practice 8
  - Best practice: Include in the test suite problems that test all levels in the hierarchy. CREATE VVUQ Practice 8
- Principle: Verification is most effective when performed on software developed under appropriate software quality practices.
  - Best practice: Use software configuration management and regression testing, and strive to understand the degree of code coverage attained by the regression suite. CREATE Core Practice 11; CREATE VVUQ Practice 4
  - Best practice: Understand that code-to-code comparisons can be helpful, especially for finding errors in the early stages of development, but that in general they do not by themselves constitute sufficient code or solution verification. CREATE VVUQ Practice 9
  - Best practice: Compare against analytic solutions, including those created by the method of manufactured solutions—a technique that is helpful in the verification process. CREATE VVUQ Practice 6
- Principle: The goal of solution verification is to estimate, and control if possible, the error in each QOI for the problem at hand. (Ultimately, of course, one would want to use UQ to facilitate the making of decisions in the face of uncertainty. So it is desirable for UQ to be tailored in a way to help identify ways to reduce uncertainty, bound it, or bypass the problem, all in the context of the decision at hand. The use of VVUQ for uncertainty management is discussed in Section 6.2.)
  - Best practice: When possible in solution verification, use goal-oriented posteriori error estimates, which give numerical error estimates for specified QOIs. In the ideal case the fidelity of the simulation is chosen so that the estimated errors are small compared to the uncertainties arising from other sources. Not addressed
  - Best practice: If goal-oriented posteriori error estimates are not available, try to perform self-convergence studies (in which QOIs are computed at different levels of refinement) on the problem at hand, which can provide helpful estimates of numerical error. CREATE VVUQ Practice 6.2

Validation and Prediction Principles and Best Practices

- Principle: A validation assessment is well defined only in terms of specified quantities of interest (QOIs).
  - Best practice: Early in the validation process, specify the QOIs that will be addressed. CREATE VVUQ Practice 9
- Principle: A validation assessment provides direct information about model accuracy only in the domain of applicability that is “covered” by the physical observations employed in the assessment.
  - Best practice: When quantifying or bounding model error for a QOI in the problem at hand, systematically assess the relevance of supporting validation assessments (which were based on data from different problems, often with different QOIs). Subject-matter expertise should inform this assessment of relevance.
- Best practice: If possible, use a broad range of physical observation sources so that the accuracy of a model can be checked under different conditions and at multiple levels of integration. CREATE VVUQ Practice 11
- Best practice: Use “holdout tests” to test validation and prediction methodologies. In such a test validation data is withheld from the validation process, the prediction machinery is employed to “predict” the withheld QOIs, with quantified uncertainties, and finally the predictions are compared to the withheld data. Not included
- Best practice: If the desired QOI was not observed for the physical systems used in the validation process, compare sensitivities of the available physical observations with those of the QOI.
- Best practice: Consider multiple metrics for comparing model outputs against physical observations. CREATE VVUQ Practice 11
- Principle: The efficiency and effectiveness of validation assessment are often improved by exploiting the hierarchical composition of computational and mathematical models, with assessments beginning on the lowest-level building blocks and proceeding to successively more complex levels.
  - Best practice: Identify hierarchies in computational and mathematical models, seek measured data that facilitates hierarchical validation assessments, and exploit the hierarchical composition to the extent possible. CREATE VVUQ Practice 10
  - Best practice: If possible, use physical observations, especially at more basic levels of the hierarchy, to constrain uncertainties in model inputs and parameters. CREATE VVUQ Practice 10
- Principle: The uncertainty in the prediction of a physical QOI must be aggregated from uncertainties and errors introduced by many sources, including: discrepancies in the mathematical model, numerical and code errors in the computational model, and uncertainties in model inputs and parameters.
  - Best practice: Document assumptions that go into the assessment of uncertainty in the predicted QOI, and also document any omitted factors. Record the justification for each assumption and omission. CREATE Practice 9
  - Best practice: Assess the sensitivity of the predicted QOI and its associated uncertainties.
Workflow Management

Our Analysis

Notional Home Ground Chart for CREATE

*after* Boehm, Using Risk to Balance Agile and Plan Driven Methods, IEEE Computer Society, 2003

Our Analysis

Personnel Experience

- Low Competency

Criticality

- High
- Low

Team Size

- Large
- High dependence on order

Culture

Requirements Dynamism

- Agile
- Plan-driven
Workflow Management

Our Approach

Milestone-driven, but flexible execution

Workflow Management

Our Approach

Iterative with Annual Releases

Canonical Milestones
- Preliminary Design Review
- Final Design Review
- New Development Branch
- Freeze of Development Branch
- Alpha Testing
- Beta Testing
- CCB assessment of readiness for release
- Release
- End-of-life for version

CMMI Best Practices for:

- Program Management
- Requirements Management
- Configuration Management
- Quality Assurance
- Verification, Validation & UQ
### AV Integrated Milestone Chart (Covering 4 Codes) -- FY2013

<table>
<thead>
<tr>
<th>Value Earning Milestone</th>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
<th>Jun</th>
<th>Jul</th>
<th>Aug</th>
<th>Sep</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDR (Preliminary Design Review)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDR (Final Design Review)</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Branch</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Requirements Reconciliation (R+3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Freeze Release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td>5</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complete (internal) Testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>PAT Release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complete PAT (Product Acceptance Test)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td>5</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BETA Release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>End of Life</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Distribution C. Please see Page 1 for additional information
CREATE Product Release Cadence

<table>
<thead>
<tr>
<th>Fiscal Year</th>
<th>FY2010</th>
<th>FY2011</th>
<th>FY2012</th>
<th>FY2013-planned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quarter</td>
<td>1 2 3 4</td>
<td>1 2 3 4</td>
<td>1 2 3 4</td>
<td>1 2 3 4</td>
</tr>
<tr>
<td>AV-DaVinci</td>
<td>1 1</td>
<td>1 2</td>
<td>2 3</td>
<td>4 3</td>
</tr>
<tr>
<td>AV-Helios</td>
<td>1</td>
<td>1 2</td>
<td>3 3</td>
<td>4 4</td>
</tr>
<tr>
<td>AV-Kestrel</td>
<td>1</td>
<td></td>
<td>2 3</td>
<td>4 4</td>
</tr>
<tr>
<td>MG-Capstone</td>
<td>1 1.5 1</td>
<td>2 2</td>
<td>3 3</td>
<td>4 4</td>
</tr>
<tr>
<td>RF-SENTRI</td>
<td>1 1.5 1</td>
<td>2 2</td>
<td>3 3</td>
<td>4 4</td>
</tr>
<tr>
<td>Ships-IHDE</td>
<td>1 1</td>
<td>2 2.1</td>
<td>3 3</td>
<td>4 4</td>
</tr>
<tr>
<td>Ships-NavyFoam</td>
<td>1 1</td>
<td>2 2</td>
<td>3 3</td>
<td>4 4</td>
</tr>
<tr>
<td>Ships-NESM</td>
<td>0.1 1</td>
<td>1 1.1</td>
<td>0.5 1.5</td>
<td></td>
</tr>
<tr>
<td>Ships-RSDE</td>
<td></td>
<td>1 1</td>
<td>1 1</td>
<td></td>
</tr>
</tbody>
</table>
Benefits of Annual Releases

- Reach Closure on Incremental Capabilities
- Provides annual demonstration of significant progress
- Creates prototypes which facilitate customer testing and input
- Mitigation for Requirements Creep

<table>
<thead>
<tr>
<th>NESM Version</th>
<th>Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>NESM v0.1</td>
<td>• Preliminary UC I</td>
</tr>
<tr>
<td>NESM v1.0</td>
<td>• Verified UC I</td>
</tr>
<tr>
<td></td>
<td>• Preliminary UC II</td>
</tr>
<tr>
<td>NESM v1.1</td>
<td>• Partially Validated UC I</td>
</tr>
<tr>
<td></td>
<td>• Verified UC II</td>
</tr>
<tr>
<td>NESM v2.0</td>
<td>• Partially Validated UC II</td>
</tr>
<tr>
<td></td>
<td>• Preliminary UC III</td>
</tr>
<tr>
<td>NESM v3.0</td>
<td>• Partially Validated UC III</td>
</tr>
<tr>
<td></td>
<td>• Preliminary UC IV</td>
</tr>
<tr>
<td>NESM v4.0</td>
<td>• Verified UC IV</td>
</tr>
<tr>
<td>NESM v4.1</td>
<td>• Partially Validated UC IV</td>
</tr>
<tr>
<td></td>
<td>• Preliminary UC V</td>
</tr>
<tr>
<td>NESM v5.0</td>
<td>• Partially Validated UC V</td>
</tr>
<tr>
<td></td>
<td>• Preliminary UC VI</td>
</tr>
<tr>
<td>NESM v6.0</td>
<td>• Partially Validated UC VI</td>
</tr>
</tbody>
</table>

Note: Startups may take longer
Shared Development Practices

- Requirements Management
- Software Quality Attributes
- Design & Implementation
- Software Configuration Management
- Verification & Validation
- Release Practices
- Customer Support

Support only 2 releases
Issue Tracker (JIRA)
CREATE Community Web Services
User Forums (CREATE Forum)
On-line Application Documentation

Maintainability
Extensibility
Performance

Workflow Management:
Agile, Iterative, Spiral

Central code repository
Configuration Management Tools (Subversion)
Document Repository (Confluence)
Configuration Control Boards

Annual Releases
### Well-Documented Software Development Plans

**Capstone Backlog**

#### Work Breakdown

<table>
<thead>
<tr>
<th>Feature/Capability</th>
<th>Implementing Work Package</th>
<th>Sub-package (Backlog)</th>
<th>Dependencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometry Database</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geometry Basic API enhancements</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geometry Query API enhancements</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geometry Modify API enhancements</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geometry AdvModify API enhancements</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geometry Visualization API enhancements</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology Data Structure (on the top of standard kernel topo)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology Implementation in SMLIB Geometry Database</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology Implementation in Discrete Geometry Database</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology implementation in Parasolid Geometry Database</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Handling of Automatic Volume Creation Issue in SMLIB (via Super-Topology?)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Composite patch reparametrization</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology base topology discrete parametrization via unstructured triangular meshes &amp; segmented edges</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology 2 tiers patch/base parametrization (for quick evaluation)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functionality to automatically &amp; manually define “Gap” face definition (closing boundaries etc.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>“Gap” face implementation in super-topology</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>“Gap” face (closed boundaries) using Delaunay point insertion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>“Gap” face meshing using NURBS evaluation (Itoh’s style)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interpolation scheme for patch &gt; base topologies for fast evaluation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functionality to automatically define composites based on attribution, angle threshold etc.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function to manually define composites</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology Workflow (Version #1) The super-topology is only created and maintained when generating meshes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super-Topology Workflow (Version #2) the super-topology is always maintained while working on the model</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Baseline Schedule

- **Release Capstone 1.0**: Nov 30, 2010
- **Release Capstone 2.0**: Nov 30, 2010
- **Beta-testing**: May 30, 2011
- **Augmentation**: Jul 30, 2011
- **Release Capstone 2.0**: Sep 30, 2011

---

Distribution C. Please see Page 1 for additional information.
Well-Documented Applications

- Annual Project Baseline
- Application Technical Description
- Annual Software Development Plan
- Developer’s Guide
- User’s Guide
- Test Plan
- Test Report
Prototype: A Example from MG- Capstone

- Original Requirement: Non-penetrating component implant (MG-09-UC-01)
  1: non-penetrating implant (imprint) [MG-09-UC-01]
    a) exact vertex imprint (recover designated vertices of the component in the ship mesh)
    b) exact edge imprint (recover designated vertices and edges of the component in the ship mesh)
  2: penetrating implant (boolean) [MG-09-UC-02]
    a) surface mesh only (both component and ships are surface meshes) (second slide)
    b) mixed-dimension (component and ships may have mixture of edge/face/region entities)
Prototypes: An Example from MG-Capstone

• Penetrating component implant (MG-09-UC-02)
MG Prototype Cadence

<table>
<thead>
<tr>
<th>Nov</th>
<th>Dec</th>
<th>Mar</th>
<th>Jul</th>
<th>Aug</th>
<th>Nov</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release n-1</td>
<td>Requirements Reconciliation and PDR n</td>
<td>FDR n</td>
<td>Alpha n</td>
<td>Beta n</td>
<td>Release n</td>
</tr>
</tbody>
</table>

Prototype development

CCB Events
Summary

CREATE has successfully managed risk with sound software Development practices --

1. Based on lessons learned from real scientific code projects [from DARPA HPCS Case Studies and others of both successes and failures]

2. Addressing documented risks inherent in these projects [based on Software Engineering Institute Risk Taxonomy]

3. Documented in CREATE Software Development Guidance [SEPP and PMP]

4. Resulting in 100% success to date
Backup Slides
CREATE Development Rhythm

Concept Development
1-5 Years
- Initial Conceptual Design & Roadmap
- Annual Multi-Physics Integration Prototyping Loop

Software Development
6-9 Years
- Finalized Design
- Annual Performance Scaling Prototype Loop

Deployment and Support
10-12 Years
- Adaptation to Acquisition Workflows Loop

Key Concepts:
- Physics Integration
- Performance and Usability Enhancements
- Deployment and Customer-driven Enhancements

Key concept: Physics Integration

Approved ICD

Have all of the Multi-Physics Integration issues been resolved and prototyped?

Have all of the Performance Thresholds been reached?

Have all the Customer Requirements been addressed?

Distribution C. Please see Page 1 for additional information
Flexible Workflow Execution

The Management of CREATE Development Workflow

Iterative Workflow

- **Iterative (and Incremental)**
  - All
- **Agile (Scrum-like)**
  - RF, Capstone, NESM, NavyFoam, RSDI, IHDE, DaVinci
- **Spiral**
  - Capstone
CREATE Canonical Milestones (from Annual Software Engineering Plan)

A. *Baseline Schedule:* Includes design question milestones, if applicable. This schedule should conform to the guidance provided in the Guidance for Product Development Measurement. This schedule must include the software development milestones listed below for each version of the product under active development or support (illustrated in Figure 2) during the fiscal year of the plan:

b. Completion of Final Design Review.
c. Creation of a new development branch of the program library for the annual development cycle (alternatively, spinoff of production release branch with all development in the trunk)
d. Freeze of the product development branch to new product features.
e. End of alpha testing.
f. Completion of beta testing.
g. Completion of readiness assessment for production release (including version requirements reconciliation).
h. Completion of production release.
i. End of life for version.
CMMI - Scrum Mapping: Some examples

<table>
<thead>
<tr>
<th>Requirements</th>
<th>CMMI Practice</th>
<th>Scrum Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP 1.1</td>
<td>Develop understand on meaning</td>
<td>Review Backlog with Product owner</td>
</tr>
<tr>
<td>SP 1.2</td>
<td>Obtain participant commitment</td>
<td>Sprint planning sessions that seek team</td>
</tr>
<tr>
<td></td>
<td></td>
<td>commitment</td>
</tr>
<tr>
<td>SP 1.3</td>
<td>Manage requirements changes</td>
<td>Add stories to product backlog</td>
</tr>
<tr>
<td>SP 1.5</td>
<td>Identify inconsistencies</td>
<td>Daily Stand-up meetings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sprint planning sessions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Burndown charts</td>
</tr>
</tbody>
</table>

Project Planning

<table>
<thead>
<tr>
<th>Requirements</th>
<th>CMMI Practice</th>
<th>Scrum Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP 1.1</td>
<td>Establish top-level WBS</td>
<td>Scrum backlog expanded into tasks</td>
</tr>
<tr>
<td>SP 1.2</td>
<td>Estimate work content of tasks</td>
<td>Story points (used to estimate size of</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stories)</td>
</tr>
<tr>
<td>SP 1.3</td>
<td>Define life-cycle phases</td>
<td>The Scrum Process itself</td>
</tr>
<tr>
<td>SP 2.1</td>
<td>Establish budget and schedule</td>
<td>Scrum estimates (in Ideal Time)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Estimates of work in each release</td>
</tr>
<tr>
<td>SP 2.6</td>
<td>Plan involvement of stakeholders</td>
<td>Scrum process roles (Scrum master,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Product Owner)</td>
</tr>
</tbody>
</table>
Our Customer’s Expectations

Chapter 4: Systems Engineering

Defense Acquisition Guidebook, https://dag.dau.mil
A Software Engineering History of CREATE:

ICDs
- Software Engineering Practices, v 0.1 (FY07)
- Software Engineering Practices, v 1.0 (FY08)
- Software Engineering Practices, v 2.0 (FY09)

Startup Planning
- Recruit Development Teams

Recruit PMs

Development
- Project Management Plan, v 1.0 (FY10)
- Initial Releases (FY11)

Initial Releases
- All tools have had at least 1 release; 8 two or more Releases (FY12)

3 Releases for Some CREATE Tools