Biographies

• Jeff Dutton
  – Technical Director for Jacobs Sverdrup’s Information Technology Support Services
  – Experience in software project management, systems and software process improvement, systems and software engineering, weapons systems modeling and simulation, operations research, test and evaluation, and systems and software acquisition
  – Member of the CMMI® Product Team
  – Authored a section of *CMMI Distilled: A Practical Introduction to Integrated Process Improvement*
  – SEI Visiting Scientist
  – Candidate SCAMPISM Lead Appraiser

• Rich McCabe
  – Principal member of the technical staff at the Systems and Software Consortium (formerly the Software Productivity Consortium)
  – Co-authored the Consortium’s Object-Oriented Approach to Software-Intensive Systems (OOASIS) methodology
  – Currently working on the Consortium’s Disciplined Agility (integrates agile development with the CMMI)
  – Headed the Consortium’s pioneering work in the product-line approach for systematic reuse
  – Nearly 15 years of software and system development experience with Bell Laboratories and other firms
This workshop reflects the opinions of the authors, and does not necessarily reflect a position of the Systems and Software Consortium, Jacobs Sverdrup, or the Software Engineering Institute.
Workshop Agenda

• Define the problem and set the context
• Review concepts of agile development
• Review concepts of lean software development
• Investigate applicability and usefulness of CMMI® model suite in agile/lean development efforts
• Develop summary conclusions
Valuation Approach

- **Gate 1:** Does CMMI® model suite ALLOW agile/lean dev?
  - Structural flexibility
  - Process areas
  - Goals
  - Practice flexibility
- **Gate 2:** Does the model suite SUPPORT agile/lean dev?
  - Structural sufficiency
  - Process area sufficiency
  - Goal sufficiency
  - Practice sufficiency
- **Gate 3:** Does the model suite ENHANCE agile/lean dev?
- **Gate 4:** Does agile/lean ENHANCE the model suite?
Quick Poll of Workshop Participants

• Agile development
  – How many of you are familiar with it?
  – How many of you have done agile development?

• Lean development
  – How many of you are familiar with it?
  – How many of you have done agile development?

• CMMI
  – How many of you are familiar with it?
  – How many of you are appraisers?
Problem and Context

- Define the problem and set the context
- Review concepts of agile development
- Review concepts of lean software development
- Investigate applicability and usefulness of CMMI® model suite in agile/lean development efforts
- Develop summary conclusions
The Problem

Effective approaches to developing complex software-intensive systems

- Software Intensive System—relies on software to provide core or priority mission capability
- Typical attributes of SIS development projects
  - Large team (tens to hundreds of developers)
  - Long schedule (months to years)
  - High cost and commitment ($M)
  - Composed of multiple systems or subsystems, all or most of which contain software
  - Often incorporate many off-the-shelf components
Challenges of SIS Development

• Software requirements
  – Vague and subtle, representing subjective tradeoffs; difficult to discover and pin down “in full”
  – Volatile, responding to budget and mission changes
  – Interdependent with solution concept and design tradeoffs

• Software design
  – Complex with many degrees of freedom
  – Architecture sensitive to detailed design tradeoffs

• Integration and communication
  – Coordination across groups often slow, dysfunctional
  – Test and integration often unpredictable, interminable
Agile Development

- Define the problem and set the context
- **Review concepts of agile development**
- Review concepts of lean software development
- Investigate applicability and usefulness of CMMI® model suite in agile/lean development efforts
- Develop summary conclusions
Workshop Discussion

What are the important attributes of an agile development effort?
What Is Agile Development?

• Evolving systems in short iterations
  – Each release is a working system
  – Design for change
  – Focus on value
  – Actively guide to convergence

• Communicating efficiently

• Leveraging human strengths
  – Engage, align, and empower the team
  – Get power from each member

Comparing various interpretations of agile development, these themes seem to be common and essential (and non-specific to software)
Agile Manifesto*

We believe in practices that emphasize

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

* Paraphrased from “Manifesto for Agile Software Development” at www.agilealliance.org
Agile Principles*

• First and foremost: Satisfy the customer — Deliver working, valuable software early and frequently
• Measure progress primarily by working software
• Have business people and developers work together daily
• Welcome changing requirements
• Create a self-organizing team of motivated individuals
• Communicate using face-to-face conversation
• Avoid nonessential work
• Maintain a sustainable pace of development
• Attend continuously to good design
• Retrospect and adjust regularly

* Paraphrased from “Principles Behind the Agile Manifesto” at www.agilealliance.org/principles.html
Agile “Brand Name” Methodologies

• eXtreme Programming (XP) [Beck]
  – Widest known, developer-focused for small teams

• Crystal methodologies [Coburn]
  – Set of methodologies conditional on circumstances—
    Only 2 defined: Crystal Clear, Crystal Orange

• Feature-Driven Development (FDD) [Palmer]
  – Agile approach closest to conventional development

• Scrum [Schwaber]
  – Focused on management practices

• Lean Software Development [Poppendieck]
  – Inspired by Toyota Production System, particularly its
    product development practices
Crystal Methodologies*

• Crystal is a family of agile methodologies characterized by
  – Priorities
  – Principles
  – Properties
    • Frequent delivery
    • Reflective improvement
    • Close communication
    • Personal safety
    • Focus
    • Easy access to expert users
    • Automated testing, CM, and frequent integration
  – Strategies and techniques in practice
• Crystal methodologies vary by project size and criticality
  – Crystal Clear is the most tolerant process for a small team

* Paraphrased from Crystal Clear by Alistair Cockburn
XP Core “Xtudes” (Core Techniques)*

- Fine scale feedback
  - Test-driven development via programmer tests and customer tests
  - Planning game
  - Whole team
  - Pair programming
- Programmer welfare
  - Sustainable pace
- Shared understanding
  - Simple design
  - System metaphor
  - Collective code ownership
  - Coding standard or coding conventions
- Continuous process rather than batch
  - Continuous integration
  - Design improvement / refactoring
  - Small releases

**FDD* Processes**

- Select domain experts, chief programmers and the chief architect
- Develop an overall model
  - What classes are in the domain, how are they connected to one another and under what constraints
- Build a features list
  - For each subject area, a list of the business activities
- Plan by feature
  - Development plan with completion dates and assignments
- Design by feature
  - Inspected design package
- Build by feature

* [http://www.featuredrivendevelopment.com/](http://www.featuredrivendevelopment.com/)
Scrum

- Agile process to manage and control development work
  - Work from a backlog of prioritized features
  - Deliver in 30-day sprints
  - Coordinate via 15-minute daily status meeting
- Wrapper for existing engineering practices
- Oriented to rapidly-changing requirements
- Controls the chaos of conflicting interests and needs
- Maximizes productivity, communications, and cooperation — detects and removes obstacles to project success
- Scalable from single projects to entire organizations
- Want everyone to feel good about their job and their contributions

* Paraphrased from [http://www.controlchaos.com/about/](http://www.controlchaos.com/about/)
Typical Agile Development

• Applications evolve in multiple short iterations
  – Iterations are constant length, in range of 2-13 weeks
  – Release a working application at end of each iteration
  – Add as many of customer’s highest priority features to each new release as can fit in an iteration
  – Requirements and design elaborated each release to support features in that release
  – Extensively test features in each iteration

• Customer (or customer surrogate) reviews each release—can redirect priorities for next iteration
• Track project progress by features completed
• Never slip a release date, instead slip features
A Typical Agile Process Depiction

- **Envision & Prepare**
- **Adjust & Predict Iteration**
- **Develop Iteration**
- **Demo & Retrospect**
- **Deploy & Support**

- System sliced vertically, evolved iteratively

- Management / Governance

- 2-13 week iterations
Elaborate requirements & develop acceptance test

Update baseline & run all unit tests

Minutes

Update baseline & run all acceptance tests

Hours

Create or fix test

Individual or pair work cycle

Develop or fix asset

Runs “acceptance tests” to integrate with rest ongoing analysis work

Run test

Check in/check out cycle integrates with rest of team

Typical Loops Within Develop Iteration
A Conventional Waterfall Process

Management / Governance

Explore

Concept & Commit

System Requirements Definition

Subsystem Definition

Component Definition

Component Detailed Design

Component Coding

Component Design

Subsystem Design

Subsystem Integration & Verification

System Integration & Verification

System Architecture

Validation

Deploy & Support

System, process, and organization are sliced horizontally

Recursion
Rough Mapping: Waterfall to Agile

Envision & Prepare
- Explore Concept & Commit
- Initial requirements & architecture

Adjust & Predict Iteration
- Elaborate select parts of requirements & architecture
- 2-13 week iterations

Develop Iteration
- Most of the V but not in V-style

Demo & Retrospect
- Reflect & improve (not in the waterfall model)

Management/ Governance
- Validation

Deliver & Support
- Deploy & Support

Most of the V but not in V-style
Typical and Possible Agile Practices

- Automated testing
- Barely sufficient documentation
- Bottleneck management
- Coding standards
- Collective code ownership
- Colocation
- Continuous team integration and CM
- CRC cards
- Customer focus group review
- Customer onsite

- Daily standup
- Design metaphor
- Exploratory spikes
- Feature-based planning
- Group design
- Information radiators
- Inspections
- “Intentional” design
- Issue tracking
- Monitor and adjust
- Pair programming
- Project velocity
- Refactoring

- Retrospectives
- Risk management
- Self-tasking
- Simple, robust design
- Small releases
- Sustainable pace
- Test-driven development
- Test first
- Unit testing
- Unity statement
- Use cases
- User stories
Potential Agile Benefits

- More predictable deliveries
- Early return on investment; working software delivered and in use sooner
- Quick response to changes in customer needs
- Risk mitigation provided by shorter delivery cycles
  - Multiple opportunities to recover from missteps
  - Validation of requirements
  - Confirmation of technical approach
  - Realistic assessment of progress
- High productivity and quality
- Satisfied customers, successful projects
Lean Software Development

- Define the problem and set the context
- Review concepts of agile development
- **Review concepts of lean software development**
- Investigate applicability and usefulness of CMMI® model suite in agile/lean development efforts
- Develop summary conclusions
What are the important attributes of a lean development effort?

How does lean differ from agile development?
Lean SW Development

- Quality Redefined
- User/Customer Involvement
- The Idea of Iterations
- Iteration Management and Convergence
- Options Thinking
- Decide as Late as Possible
- Deliver as Fast as Possible
- Tacit Knowledge (vs. Process) and Rapid Learning
- Concurrency and Communication (IPT)
- Agile Engineering Support
- Lean/Agile Project Management
- Waste in Lean/Agile Development
Lean SW Development

- Quality Redefined
- User/Customer Involvement
- The Idea of Iterations
  - Iteration Management and Convergence
  - Options Thinking
  - Decide as Late as Possible
  - Deliver as Fast as Possible
  - Tacit Knowledge (vs. Process) and Rapid Learning
  - Concurrency and Communication (IPT)
  - Agile Engineering Support
  - Lean/Agile Project Management
  - Waste in Lean/Agile Development
Quality Redefined

• Variation is not (necessarily) bad
  – (Too) detailed processes can be restrictive
  – Software development is a creative process
• “Do it right the first time” is a BAD idea
  – Fast development drives out the “right” requirements
  – Fast development produces mistakes – which are the (very) basis for learning, product (quality) and value
User/Customer Involvement

- (Near) continuous feedback and tight coupling to the users/customer is a hard requirement of lean/agile development
- User/customer “awakening” occurs over several iterations of the software
- Lack of user/customer coupling drastically reduces effectiveness of lean/agile approach
The Idea of Iterations

• Basic idea: fast iterations drive out requirements clarity and lead to “better” code faster, and with fewer resources
• Iterations = lean “workflow”
• Iterations are not prototypes
• Fast iterations enable “decide as late as possible”
• Fast iterations enable “options thinking”
• “Fast” means days or weeks, perhaps a month or two
Lean SW Development

• Quality Redefined
• User/Customer Involvement
• The Idea of Iterations

• **Iteration Management and Convergence**
  • Options Thinking
  • Decide as Late as Possible
  • Deliver as Fast as Possible
  • Tacit Knowledge (vs. Process) and Rapid Learning
  • Concurrency and Communication (IPT)
  • Agile Engineering Support
  • Lean/Agile Project Management
  • Waste in Lean/Agile Development
Iteration Management and Convergence

• “Pure” agility carries a significant risk of “out of bounds” solutions

• Convergence relies on:
  – Reliance on software architecture as a “vision point”
  – High level design as an adjunct to SW architecture
  – Skilled practitioners
  – Project/technical leadership skills
Lean SW Development

• Quality Redefined
• User/Customer Involvement
• The Idea of Iterations
• Iteration Management and Convergence
• **Options Thinking**
• **Decide as Late as Possible**
• **Deliver as Fast as Possible**
• Tacit Knowledge (vs. Process) and Rapid Learning
• Concurrency and Communication (IPT)
• Agile Engineering Support
• Lean/Agile Project Management
• Waste in Lean/Agile Development
Options Thinking

• Idea based on root of decision making difficulties:
  – “Up front” full requirements baseline
  – Full detailed design early in life cycle
  – “Frozen” architecture

• Options include:
  – Requirements or features
  – Detailed design
  – Designing in a tolerance for change
  – Designing in acceptance for evolution
  – Many others
Decide as Late as Possible

- Delaying decisions to the “last responsible moment” = high business value
- Depth-first approaches force premature low-level decisions
- Requirements development
  - Early decisions based on “criticality”
    - Hard-to-do’s
    - Technical challenges
    - High priority user needs
  - Spiral (sprint) requirements decisions evolve as the learning curve accelerates
- Early architecture decisions are necessary
  - Technical constraints
  - Critical user needs
  - System design constraints
Deliver as Fast as Possible

- Fast delivery forces fast coding
- Fast delivery enables delayed decisions
- Fast delivery requires near-continuous integration
- Fast delivery requires near-continuous testing (drives out defects early)
- Fast delivery enables faster delivery of high value, high quality products at less cost
- Fast delivery leads to “steady state” workflow (and to efficiency and productivity increases)
Lean SW Development

- Quality Redefined
- User/Customer Involvement
- The Idea of Iterations
- Iteration Management and Convergence
- Options Thinking
- Decide as Late as Possible
- Deliver as Fast as Possible
- Tacit Knowledge (vs. Process) and Rapid Learning
- Concurrency and Communication (IPT)
- Agile Engineering Support
- Lean/Agile Project Management
- Waste in Lean/Agile Development
Tacit Knowledge and Rapid Learning

• Tacit Knowledge = project/domain/skills knowledge in the heads of team members
• Balance of tacit knowledge with training and defined process is key
• Lean/agile development mandates a rapid learning environment
  – Skills
  – Domains
  – Technologies
  – Improvement to high-level (lean) processes
Concurrent & Communication

• Lean/agile development = crucible for concurrency and communication
• Concurrency = all team members and stakeholders have near-real-time “push” access to all project information
• Continuous push communication is critical
  – Technologies
  – Communication skill set
Lean SW Development

- Quality Redefined
- User/Customer Involvement
- The Idea of Iterations
- Iteration Management and Convergence
- Options Thinking
- Decide as Late as Possible
- Deliver as Fast as Possible
- Tacit Knowledge (vs. Process) and Rapid Learning
- Concurrency and Communication (IPT)

**Agile Engineering Support**
- Lean/Agile Project Management
- Waste in Lean/Agile Development
Agile Engineering Support

• Engineering support = CM, QA, Metrics
  • Agile Configuration Management
    – Agile check-in/check-out
    – Agile status accounting and configuration audits
    – Agile CM system
    – Agile change management
  • Agile Quality Assurance
    – Add value by reducing risk or defects in hours or a day
    – Tight coupling to project activities
  • Agile Metrics
    – Kanban or “pull” visualization for all team members
    – Project progress and design convergence
Lean SW Development

- Quality Redefined
- User/Customer Involvement
- The Idea of Iterations
- Iteration Management and Convergence
- Options Thinking
- Decide as Late as Possible
- Deliver as Fast as Possible
- Tacit Knowledge (vs. Process) and Rapid Learning
- Concurrency and Communication (IPT)
- Agile Engineering Support
- **Lean/Agile Project Management**
- Waste in Lean/Agile Development
Lean/Agile Project Management

• **(NOT) “Plan based” approaches (like “traditional” CMMI) skills:**
  – Early detailed planning
  – Early requirements “understanding” and stability
  – Focused on project monitoring against the plan

• **Lean/Agile Project Management skills:**
  – Seeing waste
  – Value stream mapping
  – Feedback
  – Iteration leadership/management
  – Options thinking
  – Last responsible moment decision making
  – Pull/Kanban systems and measurements
  – Cost of delay awareness
  – Self determination/team empowerment
  – Motivation and leadership
  – Technical expertise
  – Refactoring (design against more stable architecture)
Lean SW Development

- Quality Redefined
- User/Customer Involvement
- The Idea of Iterations
- Iteration Management and Convergence
- Options Thinking
- Decide as Late as Possible
- Deliver as Fast as Possible
- Tacit Knowledge (vs. Process) and Rapid Learning
- Concurrency and Communication (IPT)
- Agile Engineering Support
- Lean/A agile Project Management
- **Waste in Lean/A agile Development**
Waste in Lean/Agile Development

- Partially done work
- Extra processes
- Extra features
- Task switching
- Waiting
- Motion
- Defects
- Traditional oversight/control activities
CMMI Interpretation

• Define the problem and set the context
• Review concepts of agile development
• Review concepts of lean software development
• Investigate applicability and usefulness of CMMI® model suite in agile/lean development efforts
• Develop summary conclusions
Can the CMMI® model suite be applied to agile/lean development organizations?

What problems or issues (or roadblocks) might arise?
Previous Mapping Efforts

- **Agile+ (AgileTek)**
  - Extended XP to meet CMMI Level 3
- **Microsoft Solutions Framework**
  - Methodology, management training, and tool
  - Version 4 was agile “with some overhead” to achieve CMMI Level 3 consistency
- **ASCEND (BAE Systems)**
  - Variant of agile development for small project team
  - Uses Fagan inspections, Earned Value tracking
  - Claims CMMI Level 5 compatibility
Model Components

• What model components are **required**?
  – Specific goals
    (the actual goal – not the title or explanatory information)
  – Generic goals
• What model components are **expected**?
  – Specific practices
  – Generic practices
• What model components are **informative**?
  – Subpractices
  – Typical work products
  – Discipline amplifications
  – GP elaborations
  – Goal and practice titles
  – Goal and practice notes
  – References
Specific and Generic Goals

- **Required**: Specific goals and generic goals are required model components. These components must be achieved by an organization’s planned and implemented processes. Required components are essential to rating the achievement of a process area. Goal achievement (or satisfaction) is used in appraisals as the basis upon which process area satisfaction and organizational maturity are determined. *Only the statement of the specific or generic goal is a required model component.* The title of a specific or generic goal and any notes associated with the goal are considered informative model components.

*CMMI SE/SW V1.1*
Specific and Generic Practices

• **Expected**: Specific practices and generic practices are expected model components. Expected components describe what an organization will typically implement to achieve a required component. Expected components guide those implementing improvements or performing appraisals. *Either the practices as described, or acceptable alternatives to them, are expected to be present in the planned and implemented processes of the organization before goals can be considered satisfied.* **Only the statement of the practice is an expected model component.** The title of a practice and any notes associated with the practice are considered informative model components.

*CMMI SE/SW V1.1*
Informative Elements

• Informative*: Subpractices, typical work products, discipline amplifications, generic practice elaborations, goal and practice titles, goal and practice notes, and references are informative model components that help model users understand the goals and practices and how they can be achieved. *Informative components provide details that help model users get started in thinking about how to approach goals and practices.

*CMMI SE/SW V1.1
Agile/Lean Interpretation of the CMMI

Challenge Everything
# CMMI Process Areas

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ML 5</strong></td>
<td></td>
<td></td>
<td>CAR</td>
</tr>
<tr>
<td>OID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ML 4</strong></td>
<td>OPP</td>
<td>QPM</td>
<td></td>
</tr>
<tr>
<td>OPP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPF</td>
<td>IPM</td>
<td>RD</td>
<td>DAR</td>
</tr>
<tr>
<td>OPD</td>
<td>RSKM</td>
<td>TS</td>
<td></td>
</tr>
<tr>
<td>OT</td>
<td></td>
<td>PI</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ver</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Val</td>
<td></td>
</tr>
<tr>
<td><strong>ML 3</strong></td>
<td></td>
<td>REQM</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ML 2</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Process Areas**

- Project Planning
- Project Monitoring and Control
- Supplier Agreement Management
- Integrated Project Management
- Risk Management
- Quantitative Project Management
- Requirements Management
- Requirements Development
- Technical Solution
- Product Integration
- Verification
- Validation
- Measurement and Analysis
- Process and Product Quality Assurance
- Configuration Management
- Decision Analysis and Resolution
- Causal Analysis and Resolution
- Organizational Process Focus
- Organizational Process Definition
- Organizational Training
- Organizational Process Performance
- Organizational Innovation and Deployment

Process Area Valuation Approach

- **Goal Level Insufficiency**
  - Goals do not allow or support conduct of accepted lean/agile practices
- **Goal Level Sufficiency:**
  - Goals allow or support conduct of accepted lean/agile practices
  - One or more specific practices must be replaced with one or more alternative practices to support conduct of lean/agile practices
- **Practice Level Sufficiency:**
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely unhelpful
- **Informative Element Level Sufficiency:**
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely helpful
Process Area Valuation Approach

- **Goal Level Insufficiency**
  - Goals do not allow or support conduct of accepted lean/agile practices

- **Goal Level Sufficiency**:
  - Goals allow or support conduct of accepted lean/agile practices
  - One or more specific practices must be replaced with one or more alternative practices to support conduct of lean/agile practices

- **Practice Level Sufficiency**:
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely unhelpful

- **Informative Element Level Sufficiency**:
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely helpful
Practice Valuation Approach

• **Alternative practice required**
  – Practice does not allow or support conduct of accepted lean/agile practices – Alternative practice is required

• **Supportive:**
  – Practice, as stated, fully supports conduct of accepted lean/agile practices
  – Informative elements are largely unhelpful

• **Enabling:**
  – Practice, as stated, fully supports conduct of accepted lean/agile practices
  – Informative elements are largely helpful
### Overview of CMMI to Agile/Lean Match

#### Note heavy focus on plan-based process areas

<table>
<thead>
<tr>
<th>PP</th>
<th>PMC</th>
<th>SAM</th>
<th>IPM</th>
<th>RSKM</th>
<th>IT</th>
<th>ISM</th>
<th>QPM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Note extent of process areas for requirements development and management, and verification

<table>
<thead>
<tr>
<th>REQM</th>
<th>RD</th>
<th>TS</th>
<th>PI</th>
<th>VER</th>
<th>VAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Engineering Support

<table>
<thead>
<tr>
<th>CM</th>
<th>PPQA</th>
<th>MA</th>
<th>DAR</th>
<th>OEI</th>
<th>CAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Process Management

<table>
<thead>
<tr>
<th>OPF</th>
<th>OPD</th>
<th>OT</th>
<th>OPP</th>
<th>OID</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Detailed project plans and oversight against project attributes presumed stable**

**Engineering support process areas are highly developed consistent with plan-based approach**
Apparent Areas of Friction

• Empowerment and trust versus micromanagement
  – Process and Product Quality Assurance
• Organization standards versus project standards
  – Quantitative Project Management
  – All the “Organizational” process areas
• Elaboration and review of intermediate work products
  – Requirements Management
  – Requirements Development
  – Technical Solution
  – Verification
Empowerment and Trust

• Agile/Lean enhances productivity by empowerment (team and each member has both responsibility and authority)
  – Bottom-line results of each iteration provide external accountability across iterations
  – Peer pressure provides internal accountability
  – Improvements in process are a team responsibility
• External audits undercut this agile/lean philosophy
  – QA is independent—self-discipline is demotivated
  – Auditing is non-value-added, justified only by lack of trust
  – Compliance becomes the focus, not effective practices justified by results
• Agile Coach is a hybrid role—challenges team behaviors but does not dictate resolutions—can QA become a coach?
Organization Versus Project Standards

- Agile/Lean teams determine their own process and practices by consensus
- Does CMMI tailoring guidance allow project team data or consensus to overrule
  - Organizational standards?
  - Accumulated organizational performance data?
- Otherwise, process performers are no longer the process owners
  - See previous discussion of empowerment and trust
Intermediate Work Products

• Agile/Lean suspects any non-deliverable is waste
  – Code is a necessary “detailed spec” for executable delivery
  – Tests drive code development, define and verify requirements
  – But conventional requirements and design docs only support understanding, hence “barely sufficient” documentation

• Does CMMI demand “complete” system representations in intermediate work products?
  – How much is enough to “define” and “elaborate”…
    • Requirements before …
    • Design and interfaces before …
    • Implementation and testing
  – What is sufficient review?
  – Is bi-directional traceability necessary? To what level?
Project Management
Project Planning

- SG 1 Estimates of project planning parameters are established and maintained.
- SG 2 A project plan is established and maintained as the basis for managing the project.
- SG 3 Commitments to the project plan are established and maintained.

- Good match to agile and lean!
- However, must interpret in light of
  - Large-grained initial release plan (features roughly allocated to iterations)
  - More detailed planning to begin each iteration
  - Work Breakdown Structure likely different (distinctions between testing and development less important)
## Project Planning

### SG 1 Establish Estimates
- **SP 1.1** Estimate the Scope of the Project
- **SP 1.2** Establish Estimates of Work Product and Task Attributes
- **SP 1.3** Define Project Life Cycle
- **SP 1.4** Determine Estimates of Effort and Cost

### SG 2 Develop a Project Plan
- **SP 2.1** Establish the Budget and Schedule
- **SP 2.2** Identify Project Risks
- **SP 2.3** Plan for Data Management
- **SP 2.4** Plan for Project Resources
- **SP 2.5** Plan for Needed Knowledge and Skills
- **SP 2.6** Plan Stakeholder Involvement
- **SP 2.7** Establish the Project Plan

### SG 3 Obtain Commitment to the Plan
- **SP 3.1** Review Plans that Affect the Project
- **SP 3.2** Reconcile Work and Resource Levels
- **SP 3.3** Obtain Plan Commitment

- **Acceptable**
- **Supportive**
- **Enabling**
- **Alternative**
Project Monitoring and Control

- SG 1 Actual performance and progress of the project are monitored against the project
- SG 2 Corrective actions are managed to closure when the project's performance or results deviate significantly from the plan.

• Good match to agile and lean!
  - Progress tracked by tested, completed features
  - Plans and priorities reset with each iteration based on current information, customer’s ongoing guidance

• However, agile/lean is biased to different “corrective actions”
  - Drop features rather than slip an iteration release date
  - Original plan treated as an outdated prediction
Project Monitoring and Control

SG 1  Monitor Project Against Plan
   - SP 1.1 Monitor Project Planning Parameters
   - SP 1.2 Monitor Commitments
   - SP 1.3 Monitor Project Risks
   - SP 1.4 Monitor Data Management
   - SP 1.5 Monitor Stakeholder Involvement
   - SP 1.6 Conduct Progress Reviews
   - SP 1.7 Conduct Milestone Reviews

SG 2  Manage Corrective Action to Closure
   - SP 2.1 Analyze Issues
   - SP 2.2 Take Corrective Action
   - SP 2.3 Manage Corrective Action

Supplier Agreement Management

SG 1  Establish Supplier Agreements
   SP 1.1  Determine Acquisition Type
   SP 1.2  Select Suppliers
   SP 1.3  Establish Supplier Agreements

SG 2  Satisfy Supplier Agreements
   SP 2.1  Review COTS Products
   SP 2.2  Execute the Supplier Agreement
   SP 2.3  Accept the Acquired Product
   SP 2.4  Transition Products

- Tight coupling of suppliers
- Fast response times
- Fast, agile practices
Integrated Project Management

SG 1 Use the Project’s Defined Process
- SP 1.1 Establish the Project’s Defined Process
- SP 1.2 Use Organizational Process Assets for Planning Project Activities
- SP 1.3 Integrate Plans
- SP 1.4 Manage the Project Using the Integrated Plans
- SP 1.5 Contribute to the Organizational Process Assets

SG 2 Coordinate and Collaborate with Relevant Stakeholders
- SP 2.1 Manage Stakeholder Involvement
- SP 2.2 Manage Dependencies
- SP 2.3 Resolve Coordination Issues

SG 3 Use the Project’s Shared Vision for IPPD
- SP 3.1 Define Project’s Shared-Vision Context
- SP 3.2 Establish the Project’s Shared Vision

SG 4 Organize Integrated Teams for IPPD
- SP 4.1 Determine Integrated Team Structure for the Project
- SP 4.2 Develop a Preliminary Distribution of Requirements to Integrated Teams
- SP 4.3 Establish Integrated Teams

• Agile tailoring criteria
• Learn internally and through organization
• Tailor very fast

• Extremely rapid contribution to organization’s process

• Key to agile/lean efforts
• Close, continuously coupled coordination and collaboration

• Consider shared vision point architectures

• Consider agile team structure
• Critical for self-motivated teams

Risk Management

SG 1 Prepare for Risk Management
   SP 1.1 Determine Risk Sources and Categories
   SP 1.2 Define Risk Parameters
   SP 1.3 Establish Risk Management Strategy

SG 2 Identify and Analyze Risks
   SP 2.1 Identify Risks
   SP 2.2 Evaluate, Categorize, and Prioritize Risks

SG 3 Mitigate Risks
   SP 3.1 Develop Risk Mitigation Plans
   SP 3.2 Implement Risk Mitigation Plans

- Good fit
- Ramp up to agile/lean
- Record keeping
- Migrate to continuous
- Mitigation and action
Integrated Teaming

SG 1 Establish Team Composition
SP 1.1 Identify Team Tasks
SP 1.2 Identify Needed Knowledge and Skills
SP 1.3 Assign Appropriately Skilled Members

SG 2 Govern Team Operation
SP 2.1 Establish a Shared Vision
SP 2.2 Establish a Team Charter
SP 2.3 Define Roles and Responsibilities
SP 2.4 Establish Operating Procedures
SP 2.5 Collaborate among Interfacing Teams

- Best CMMI support for management of tacit knowledge
- Overall extremely good fit for agile/lean efforts
Integrated Supplier Management

SG 1  Analyze and Select Sources of Products
   SP 1.1  Analyze Potential Sources of Products
   SP 1.2  Evaluate and Determine Sources of Products

SG 2  Coordinate Work with Suppliers
   SP 2.1  Monitor Selected Supplier Processes
   SP 2.2  Evaluate Selected Supplier Work Products
   SP 2.3  Revise the Supplier Agreement or Relationship
Quantitative Project Management

 ✓ SG 1 Quantitatively manage using quality and process-performance objectives.
   ? SP 1.2 Select the subprocesses that compose the project’s defined process based on historical stability and capability data.

 ? SG 2 The performance of selected subprocesses within the project's defined process is statistically managed.
   ? SP 2.2 Establish and maintain an understanding of the variation of the selected subprocesses …
   ? SP 2.3 Monitor the performance of the selected subprocesses to determine their capability to satisfy their … objectives, and identify corrective action as necessary.

• Agile focus: reliably valuable results despite uncertainty and volatility—not predictability through invariance
• What subprocess in agile development should be “statistically managed”? Iterations? (E.g., feature points/iteration, or convergence)
• What “historical data”? From the project? From other projects?
Quantitative Project Management

SG 1 Quantitatively Manage the Project

- SP 1.1 Establish the Project’s Objectives
- SP 1.2 Compose the Defined Process
- SP 1.3 Select the Subprocesses that Will Be Statistically Managed
- SP 1.4 Manage Project Performance

SG 2 Statistically Manage Subprocess Performance

- SP 2.1 Select Measures and Analytic Techniques
- SP 2.2 Apply Statistical Methods to Understand Variation
- SP 2.3 Monitor Performance of the Selected Subprocesses
- SP 2.4 Record Statistical Management Data
## Summary Valuation for Project Management

### Process Area Valuation Approach

- **Goal Level Insufficiency**
  - Goals do not allow or support conduct of accepted lean/agile practices

- **Goal Level Sufficiency**:
  - Goals allow or support conduct of accepted lean/agile practices
  - One or more specific practices must be replaced with one or more alternative practices to support conduct of lean/agile practices

- **Practice Level Sufficiency**:
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely unhelpful

- **Informative Element Level Sufficiency**:
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely helpful
Engineering
**Requirements Management**

- Agile addresses consistency with lower-overhead practices
  - Acceptance tests tied to features
  - Group Design, Code/Design Standards
  - Clean Design and Refactoring
  - Collective Code Ownership
  - Continuous Integration and high level of communication among team members
- But is bi-directional traceability necessary for large projects?
  - And if so, to what level of granularity? To local team level?
Requirements Management

SG 1 Manage Requirements

- SP 1.1 Obtain an Understanding of Requirements
- SP 1.2 Obtain Commitment to Requirements
- SP 1.3 Manage Requirements Changes
- SP 1.4 Maintain Bidirectional Traceability of Requirements
- SP 1.5 Identify Inconsistencies between Project and Requirements

Supportive
- “Fuzzy set” at “SW system” level
- Clarity at iteration level

Enabling
- At each iteration
- Supportive
- Managing “fuzzy set” as design is refactored toward acceptable solution
- Tied group design, collective code ownership, and acceptance tests to features
- Very high level traceability in lean development
- Alternative
- Enabling
Requirements Development

SG 1 Develop Customer Requirements
- SP 1.1 Elicit Needs
- SP 1.2 Develop the Customer Requirements

SG 2 Develop Product Requirements
- SP 2.1 Establish Product and Product-Component Requirements
- SP 2.2 Allocate Product-Component Requirements
- SP 2.3 Identify Interface Requirements

SG 3 Analyze and Validate Requirements
- SP 3.1 Establish Operational Concepts and Scenarios
- SP 3.2 Establish a Definition of Required Functionality
- SP 3.3 Analyze Requirements
- SP 3.4 Analyze Requirements to Achieve Balance
- SP 3.5 Validate Requirements with Comprehensive Methods

- “Fuzzy set” at “SW system” level
- Prioritize at iteration level (e.g. by technical challenge, importance to customer, functional precedence)
- Validate only those accepted into iterations (inspect tests)

- Much less functional analysis needed
- What is done is at a much higher level of abstraction
## Technical Solution

<table>
<thead>
<tr>
<th>SG 1</th>
<th>Select Product-Component Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP 1.1</td>
<td>Develop Detailed Alternative Solutions and Selection Criteria</td>
</tr>
<tr>
<td>SP 1.2</td>
<td>Evolve Operational Concepts</td>
</tr>
<tr>
<td>SP 1.3</td>
<td>Select Product-Component Solutions</td>
</tr>
</tbody>
</table>

## SG 2 Develop the Design

| SP 2.1 | Design the Product or Product Component |
| SP 2.2 | Establish a Technical Data Package |
| SP 2.3 | Design Interfaces Using Criteria |
| SP 2.4 | Perform Make, Buy, or Reuse Analyses |

## SG 3 Implement the Product Design

| SP 3.1 | Implement the Design |
| SP 3.2 | Develop Product Support Documentation |

*Informative elements imply full-design-before-coding.*
Product Integration

SG 1  Prepare for Product Integration
SP 1.1  Determine Integration Sequence
SP 1.2  Establish the Product Integration Environment
SP 1.3  Establish Product Integration Procedures and Criteria

SG 2  Ensure Interface Compatibility
SP 2.1  Review Interface Descriptions
SP 2.2  Manage Interfaces

SG 3  Assemble Product Components and Deliver the Product
SP 3.1  Confirm Readiness of Product Components for Integration
SP 3.2  Assemble Product Components
SP 3.3  Evaluate Assembled Product Components
SP 3.4  Package and Deliver the Product or Product Component

Informative elements are based on a systemic approach that appears somewhat biased against agile/lean.
Verification

SG1: Preparation for verification is conducted.
SG2: Peer reviews are performed on selected work products.
SG3: Selected work products are verified against their specified requirements.

• What work products? How are they verified?
• Suppose
  – We only verify software (and hardware, and their integration) with tests … good enough?
  – The entire team participates in
    • Defining features (requirements), and then …
    • Creating the initial design in “whiteboard UML” …

Does that verify design against its requirements?
Verification

SG 1 Prepare for Verification
SP 1.1 Select Work Products for Verification
SP 1.2 Establish the Verification Environment
SP 1.3 Establish Verification Plan

SG 2 Perform Peer Reviews
SP 2.1 Prepare for Peer Reviews
SP 2.2 Conduct Peer Reviews
SP 2.3 Analyze Peer Review Data

SG 3 Verify Selected Work Products
SP 3.1 Perform Verification
SP 3.2 Analyze Verification Results and Identify Corrective Action

In general, informative elements imply highly detailed data and plan-rich verification activities.
Validation

SG 1  Prepare for Validation
  SP 1.1  Select Products for Validation
  SP 1.2  Establish the Validation Environment
  SP 1.3  Establish Validation Procedures and Criteria

SG 2  Validate Product or Product Components
  SP 2.1  Perform Validation
  SP 2.2  Analyze Validation Results

Acceptable

Enabling

Supportive

Alternative
Summary Valuation for Engineering

<table>
<thead>
<tr>
<th>Acceptable</th>
<th>Supportive</th>
<th>Supportive</th>
<th>Supportive</th>
<th>Acceptable</th>
<th>Acceptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQM</td>
<td>RD</td>
<td>TS</td>
<td>PI</td>
<td>VER</td>
<td>VAL</td>
</tr>
<tr>
<td>SG 1</td>
<td>Acceptable</td>
<td>Enabling</td>
<td>Supportive</td>
<td>Supportive</td>
<td>Enabling</td>
</tr>
<tr>
<td>SG 2</td>
<td>Enabling</td>
<td>Supportive</td>
<td>Supportive</td>
<td>Supportive</td>
<td>Acceptable</td>
</tr>
<tr>
<td>SG 3</td>
<td>Supportive</td>
<td>Supportive</td>
<td>Supportive</td>
<td>Acceptable</td>
<td>Acceptable</td>
</tr>
</tbody>
</table>

Process Area Valuation Approach

- **Goal Level Insufficiency**
  - Goals do not allow or support conduct of accepted lean/agile practices

- **Goal Level Sufficiency**
  - Goals allow or support conduct of accepted lean/agile practices
  - One or more specific practices must be replaced with one or more alternative practices to support conduct of lean/agile practices

- **Practice Level Sufficiency**
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely unhelpful

- **Informative Element Level Sufficiency**
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely helpful
Engineering Support
Configuration Management

- SG 1 Baselines of identified work products are established.
- SG 2 Changes to the work products under CM are tracked and controlled.
- SG 3 Integrity of baselines is established and maintained.

? SP 3.2 Perform configuration audits to maintain integrity of the configuration baselines.

• Good match to goals, but what about CM audits practice?
• Agile/Lean preference
  – Automated controls (check-in, nightly build/test)
  – Peer pressure to enforce practices (audits are expensive)
  – Communication supported by “barely sufficient” and non-definitive documents (agile modeling)
• Good enough for software artifacts?
• But audits still necessary for large, distributed teams?
  – More communication by documentation
By our definitions, these practices are enabling. However, informative elements to encourage agile, focused, lean CM are not present.
### Process and Product Quality Assurance

<table>
<thead>
<tr>
<th>SG 1</th>
<th>Objectively Evaluate Processes and Work Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP 1.1</td>
<td>Objectively Evaluate Processes</td>
</tr>
<tr>
<td>SP 1.2</td>
<td>Objectively Evaluate Work Products and Services</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SG 2</th>
<th>Provide Objective Insight</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP 2.1</td>
<td>Communicate and Ensure Resolution of Noncompliance</td>
</tr>
<tr>
<td>SP 2.2</td>
<td>Establish Records</td>
</tr>
</tbody>
</table>

---

**By our definitions, these practices are enabling. However, informative elements to encourage agile, focused, lean evaluation practices are not present.**

**Both the practices and the informative elements imply systemic, plan-based, monolithic resolution of problems.**
Measurement and Analysis

SG 1 Align Measurement and Analysis Activities
SP 1.1 Establish Measurement Objectives
SP 1.2 Specify Measures
SP 1.3 Specify Data Collection and Storage Procedures
SP 1.4 Specify Analysis Procedures

• Emphasis on voluminous metric data. Seems moot on all but large and complex programs.

SG 2 Provide Measurement Results
SP 2.1 Collect Measurement Data
SP 2.2 Analyze Measurement Data
SP 2.3 Store Data and Results
SP 2.4 Communicate Results
### Decision Analysis and Resolution

#### SG 1 Evaluate Alternatives

| SP 1.1 | Establish Guidelines for Decision Analysis |
| SP 1.2 | Establish Evaluation Criteria |
| SP 1.3 | Identify Alternative Solutions |
| SP 1.4 | Select Evaluation Methods |
| SP 1.5 | Evaluate Alternatives |
| SP 1.6 | Select Solutions |
Organizational Environment for Integration

SG 1 Evaluate Alternatives

SP 1.1 Establish Guidelines for Decision Analysis
SP 1.2 Establish Evaluation Criteria
SP 1.3 Identify Alternative Solutions
SP 1.4 Select Evaluation Methods
SP 1.5 Evaluate Alternatives
SP 1.6 Select Solutions
Causal Analysis and Resolution

SG 1 Determine Causes of Defects
   SP 1.1 Select Defect Data for Analysis
   SP 1.2 Analyze Causes

SG 2 Address Causes of Defects
   SP 2.1 Implement the Action Proposals
   SP 2.2 Evaluate the Effect of Changes
   SP 2.3 Record Data
Summary Valuation for Engineering Support

<table>
<thead>
<tr>
<th>Process Area Valuation Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Goal Level Insufficiency</td>
</tr>
<tr>
<td>– Goals do not allow or support conduct of accepted lean/agile practices</td>
</tr>
<tr>
<td>• Goal Level Sufficiency:</td>
</tr>
<tr>
<td>– Goals allow or support conduct of accepted lean/agile practices</td>
</tr>
<tr>
<td>– One or more specific practices must be replaced with one or more alternative practices to support conduct of lean/agile practices</td>
</tr>
<tr>
<td>• Practice Level Sufficiency:</td>
</tr>
<tr>
<td>– Goals allow or support conduct of accepted lean/agile practices</td>
</tr>
<tr>
<td>– Practices, as stated, fully support conduct of accepted lean/agile practices</td>
</tr>
<tr>
<td>– Informative elements are largely unhelpful</td>
</tr>
<tr>
<td>• Informative Element Level Sufficiency:</td>
</tr>
<tr>
<td>– Goals allow or support conduct of accepted lean/agile practices</td>
</tr>
<tr>
<td>– Practices, as stated, fully support conduct of accepted lean/agile practices</td>
</tr>
<tr>
<td>– Informative elements are largely helpful</td>
</tr>
</tbody>
</table>
Process Management
By our definitions, these practices are enabling. However, informative elements to encourage agile, focused, lean and rapid process improvement are absent.

Informative elements are contrary to rapid continuous improvement
Organizational Process Definition

SG 1 Establish Organizational Process Assets
   SP 1.1 Establish Standard Processes
   SP 1.2 Establish Life-Cycle Model Descriptions
   SP 1.3 Establish Tailoring Criteria and Guidelines
   SP 1.4 Establish the Organization’s Measurement Repository
   SP 1.5 Establish the Organization’s Process Asset Library

Life cycle informative elements are not all helpful in agile/lean efforts.

Informative elements are too focused on systemic tailoring.
Organizational Training

SG 1 Establish an Organizational Training Capability
   SP 1.1 Establish the Strategic Training Needs
   SP 1.2 Determine Which Training Needs Are the Responsibility of the Organization
   SP 1.3 Establish an Organizational Training Tactical Plan
   SP 1.4 Establish Training Capability

SG 2 Provide Necessary Training
   SP 2.1 Deliver Training
   SP 2.2 Establish Training Records
   SP 2.3 Assess Training Effectiveness

Although these are all rated as enabling, informative elements that support the identification and application of tacit knowledge are missing.
Organizational Process Performance

SG 1 Establish Performance Baselines and Models

SP 1.1 Select Processes
SP 1.2 Establish Process Performance Measures
SP 1.3 Establish Quality and Process-Performance Objectives
SP 1.4 Establish Process Performance Baselines
SP 1.5 Establish Process Performance Models

Although these are all rated as enabling, elements such as improvement of skills-based teams with highly developed tacit knowledge are missing.
SG 1  Select Improvements
- SP 1.1 Collect and Analyze Improvement Proposals
- SP 1.2 Identify and Analyze Innovations
- SP 1.3 Pilot Improvements
- SP 1.4 Select Improvements for Deployment

SG 2  Deploy Improvements
- SP 2.1 Plan the Deployment
- SP 2.2 Manage the Deployment
- SP 2.3 Measure Improvement Effects
Summary Valuation for Process Management

<table>
<thead>
<tr>
<th>SG 1</th>
<th>Supportive</th>
<th>OPF</th>
<th>Enabling</th>
<th>OT</th>
<th>Enabling</th>
<th>OPP</th>
<th>Enabling</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG 2</td>
<td>Supportive</td>
<td>OPD</td>
<td>Enabling</td>
<td>OT</td>
<td>Enabling</td>
<td>OPP</td>
<td>Enabling</td>
</tr>
</tbody>
</table>

**Process Area Valuation Approach**

- **Goal Level Insufficiency**
  - Goals do not allow or support conduct of accepted lean/agile practices

- **Goal Level Sufficiency**
  - Goals allow or support conduct of accepted lean/agile practices
  - One or more specific practices must be replaced with one or more alternative practices to support conduct of lean/agile practices

- **Practice Level Sufficiency**
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely unhelpful

- **Informative Element Level Sufficiency**
  - Goals allow or support conduct of accepted lean/agile practices
  - Practices, as stated, fully support conduct of accepted lean/agile practices
  - Informative elements are largely helpful
## Summary of Ratings

### Project Management

<table>
<thead>
<tr>
<th>SG</th>
<th>Acceptable</th>
<th>Supportive</th>
<th>Enabling</th>
<th>Supportive</th>
<th>Enabling</th>
<th>Enabling</th>
<th>Acceptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>PP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IPM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSKM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QPM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Engineering

<table>
<thead>
<tr>
<th>SG</th>
<th>Acceptable</th>
<th>Supportive</th>
<th>Supportive</th>
<th>Supportive</th>
<th>Acceptable</th>
<th>Acceptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Engineering Support

<table>
<thead>
<tr>
<th>SG</th>
<th>Enabling</th>
<th>Acceptable</th>
<th>Supportive</th>
<th>Enabling</th>
<th>Acceptable</th>
<th>Enabling</th>
<th>Enabling</th>
</tr>
</thead>
<tbody>
<tr>
<td>CM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPQA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OEI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Process Management

<table>
<thead>
<tr>
<th>SG</th>
<th>Supportive</th>
<th>Supportive</th>
<th>Enabling</th>
<th>Enabling</th>
<th>Enabling</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Generic Practices

- GP 1.1 Perform the base practices of the process area to develop work products and provide services to achieve the specific goals of the process area.
- GP 2.1 Establish and maintain an organizational policy for planning and performing the process.
- GP 2.2 Establish and maintain the plan for performing the process.
- GP 2.3 Provide adequate resources for performing the process, developing the work products, and providing the services of the process.
- GP 2.4 Assign responsibility and authority for performing the process, developing the work products, and providing the services of the process.
- GP 2.5 Train the people performing or supporting the process as needed.
- GP 2.6 Place designated work products of the process under appropriate levels of configuration management.
- GP 2.7 Identify and involve the relevant stakeholders as planned.
- GP 2.8 Monitor and control the process against the plan for performing the process and take appropriate corrective action.
- GP 2.9 Objectively evaluate adherence of the process against its process description, standards, and procedures, and address noncompliance.
- GP 2.10 Review the activities, status, and results of the process with higher level management and resolve issues.

- GP 2.2
  A plan for performing the process will have to be intelligently applied to avoid undue burden on agile/lean processes.

- GP 2.5
  Training of agile/lean teams and team members should take advantage of continuous project/organizational learning mechanisms—and support the building and extension of tacit knowledge and advanced skill sets.

- GP 2.6
  Judicious choice of what work products to place under CM. In addition, CM practices must be agile.

- GP 2.8
  Careful selection and definition of processes should make this GP helpful.

- GP 2.9
  QA of processes is helpful— if the processes are agile/lean—and the practice of QA is agile as well.
Generic Practices

- **GP 3.1** Establish and maintain the description of a defined process.

- **GP 3.2** Collect work products, measures, measurement results, and improvement information derived from planning and performing the process to support the future use and improvement of the organization’s processes and process assets.

- **GP 3.2** Care must be taken in the application of this GP in agile/lean environments. Process must be carefully selected and made lean. Support of “future use” must be immediate, to the project itself as well as other projects.

- **GP 4.1** Establish and maintain quantitative objectives for the process that address quality and process performance based on customer needs and business objectives.

- **GP 4.2** Stabilize the performance of one or more subprocesses to determine the ability of the process to achieve the established quantitative quality and process-performance objectives.

- **GP 4.2** As previously discussed, selection of processes to stabilize must be done with great care, as some agile/lean process are necessarily uncontrolled.

- **GP 5.1** Ensure continuous improvement of the process in fulfilling the relevant business objectives of the organization.

- **GP 5.2** Identify and correct the root causes of defects and other problems in the process.
Conclusions

- Define the problem and set the context
- Review concepts of agile development
- Review concepts of lean software development
- Investigate applicability and usefulness of CMMI® model suite in agile/lean development efforts
- Develop summary conclusions
CMMI Interpretation—Bottom Line

• Primarily focused on processes and practices
• Largely ignores human aspects of (exc. IT, OEI)
  – Knowledge acquisition
  – Collaboration
• Thorough and systemic treatment of
  – Technologies
  – Informational elements and relationships
  – (Very) early “full” development of requirements
• Structure of required, expected, and information elements provides a great deal of flexibility

Does CMMI Model suite ALLOW agile/lean dev?

YES
Value Added From CMMI

• Decision Analysis and Resolution is a counterpoint to agile bias toward “resolve by building”
• Organizational improvement beyond the project team (Organizational Environment for Integration, Training, Process Focus and Definition, Innovation and Deployment, and Process Performance)
• Hardware awareness—agile/lean ignore coordinating long-lead time efforts (Product Integration)
• Supplier interactions (ISM, SAM)
  – But note relevant agile/lean ideas
• Integrated Teaming and Organizational Environment for Integration are significant enablers for agile/lean efforts
• Robust set of practices ensures most are addressed in agile/lean efforts (where tendency may be to ignore or lessen effectiveness)

Thus, CMMI tends to reduce risk in agile/lean development
Value Added from Lean and Agile

- Iteration release rather than phased development
- Value of fast as possible production, work flow, and minimal Work In Progress
- Testing and continuous integration as essential drivers for implementation (and testing interleaved with other development activities)
- Waste reduction as a goal—testing and pair development as cost-effective options to inspection and review
- “Last responsible moment” decisions, options thinking, and incremental commitment [Gilb]
- Focus and synergy of technical leadership and technical management—practical concepts for engaging developers through empowerment
- Recognition and effective use of advanced skill sets
To Apply CMMI in Agile/Lean Environments

For Supportive practices, informative elements must be largely replaced with agile/lean elements.

For Acceptable practices, alternative agile/lean practices must be provided.

For Enabling practices, add agile/lean sub-practices, etc.
Workshop Discussion

Review CMMI® issues charts for closure.
References

- CMMI – Guidelines for Process Integration and Product Improvement, Mary Beth Chrissis, et al, Addison Wesley
- Lean Software Development – An Agile Toolkit, Mary and Tom Poppendieck, Addison Wesley
- Lean Thinking – Banish Waste and Create Wealth in Your Corporation, James P. Womack and Daniel T. Jones, Free Press
Contact Information

• Rich McCabe
  Systems and Software Consortium
  2214 Rock Hill Road
  Herndon, VA 20170-4227
  (703) 742-7289
  McCabe@sysemsandsoftware.org

• Jeff Dutton
  Jacobs Sverdrup ITSS
  6703 Odyssey Drive, Suite 305
  Huntsville, AL 35806
  (256) 971-5527
  jeff.dutton@jacobs.com