



Agile/ Lean Development and CMMI®



SEPG 2006

March 9th, 2006

Jeffrey L. Dutton
Richard S. McCabe

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

*While there is value in the items on the right,
the items on the left are more valuable*

* 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

* <http://www.c2.com/cgi/wiki?ExtremeProgrammingCorePractices>

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/>

Scrum

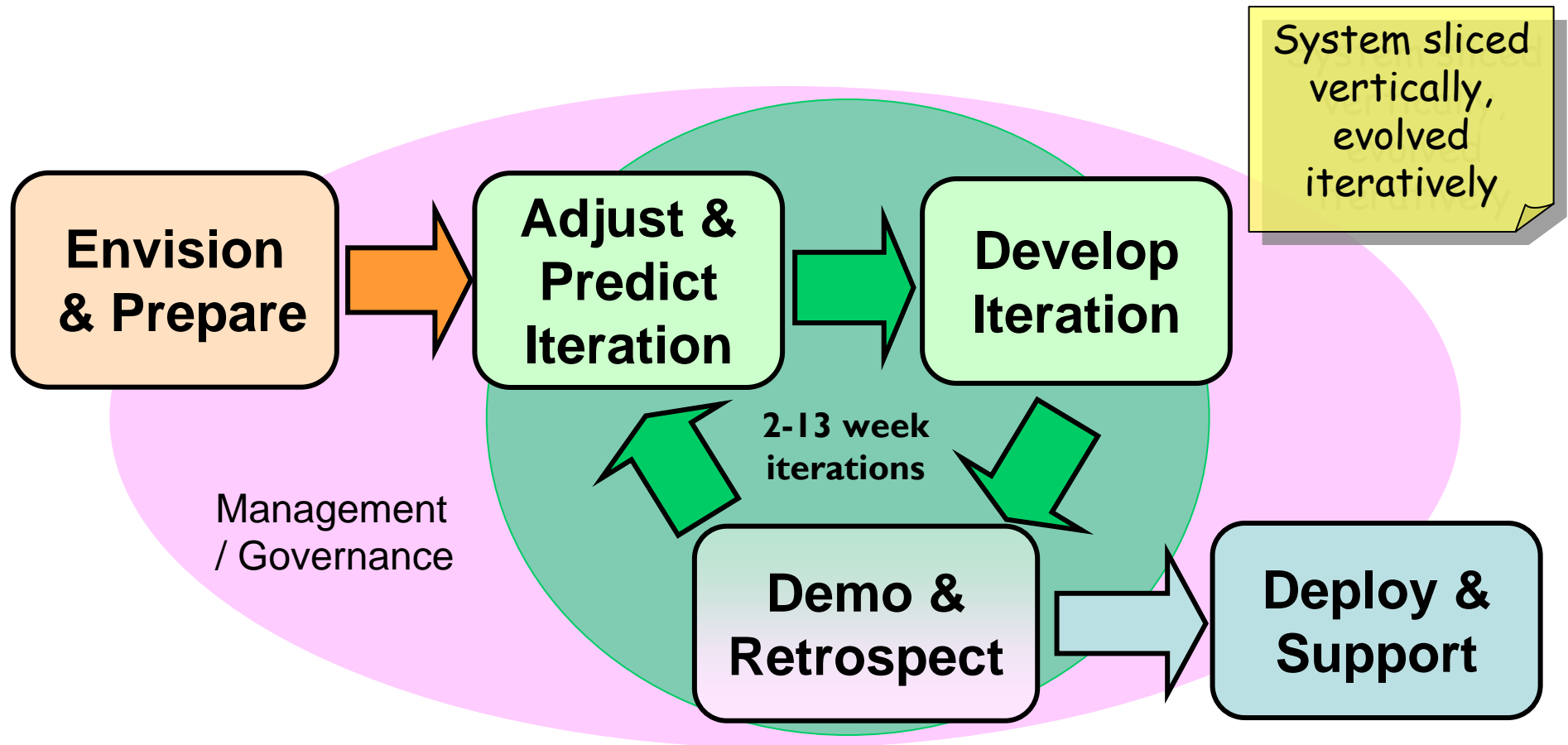
- Agile process to manage and control development work
 - Work from a backlog of prioritized features
 - Deliver in 30-day sprints
 - Coordinate via 15-minute daily status meeting
- Wrapper for existing engineering practices
- Oriented to rapidly-changing requirements
- Controls the chaos of conflicting 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/>

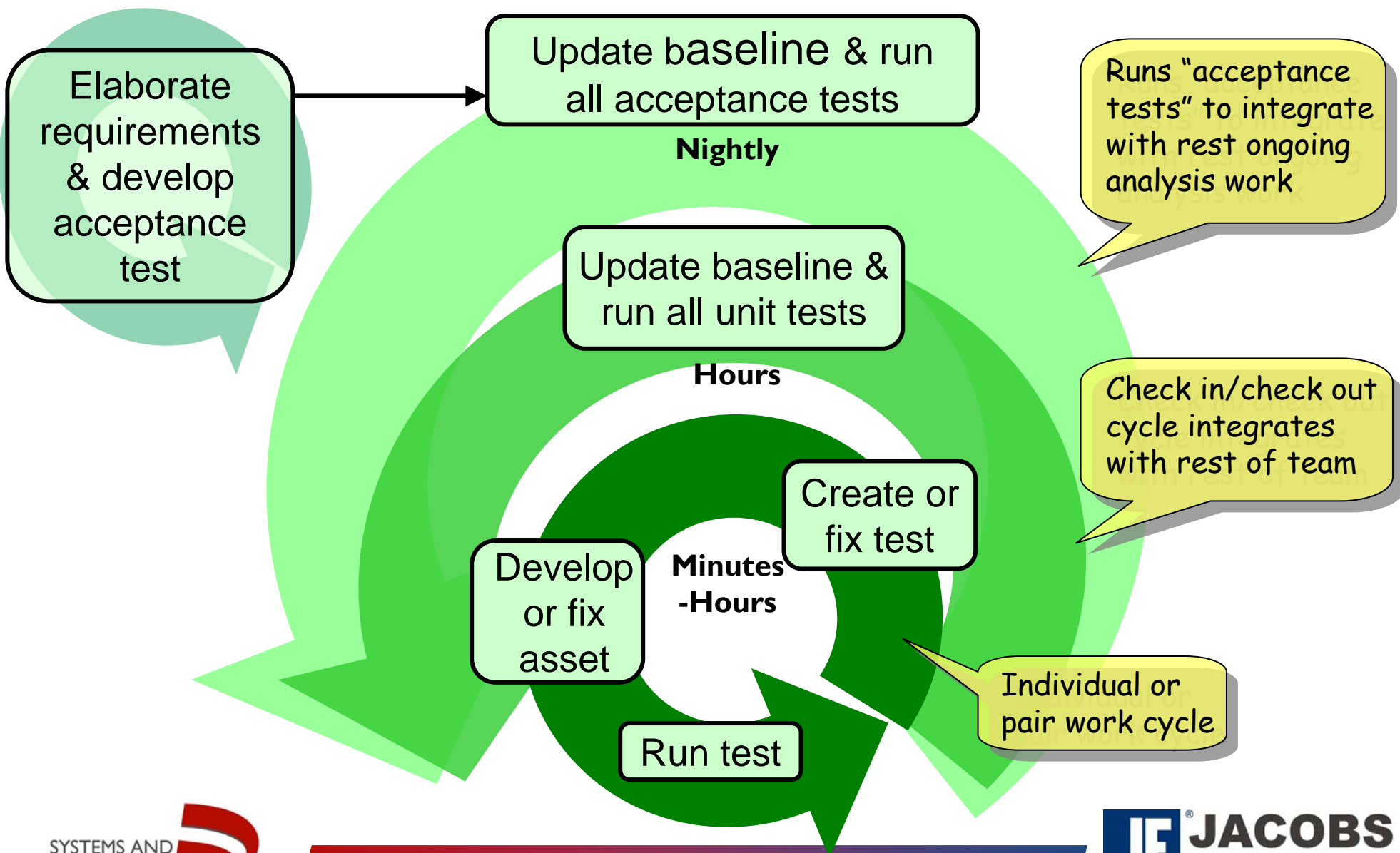
Typical Agile Development

- 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

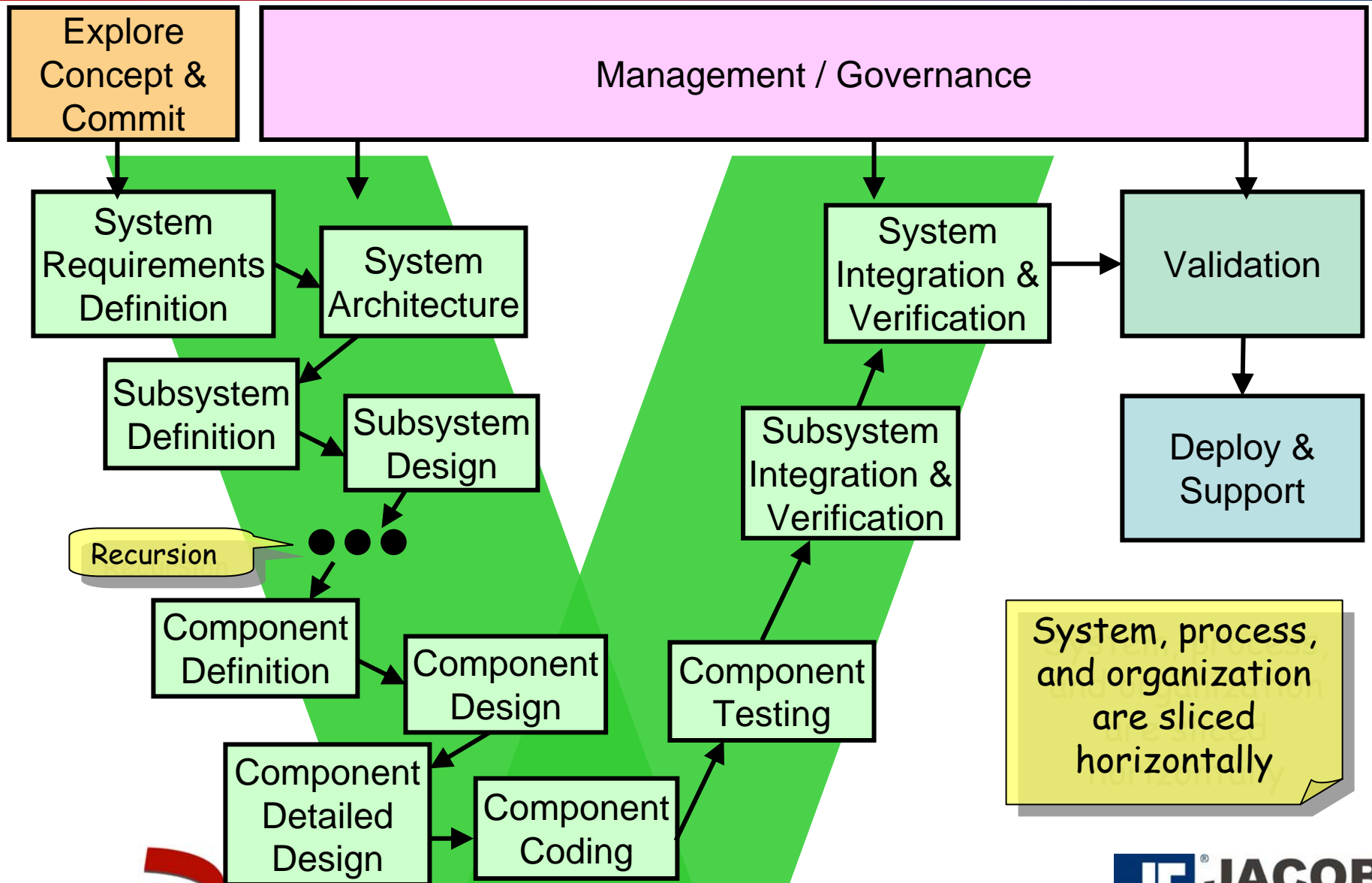
A Typical Agile Process Depiction



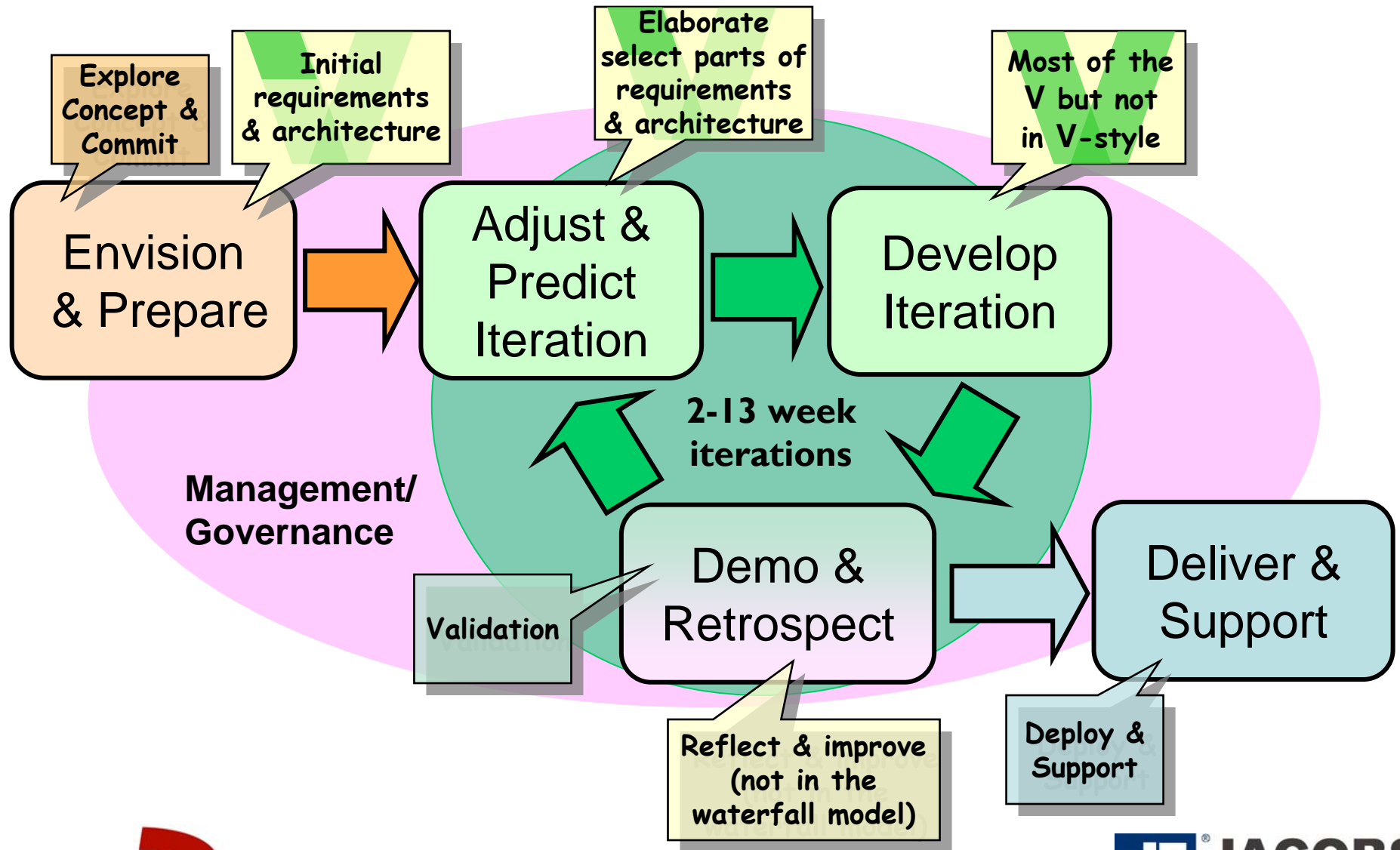
Typical Loops Within *Develop* Iteration



A Conventional Waterfall Process



Rough Mapping: Waterfall to Agile

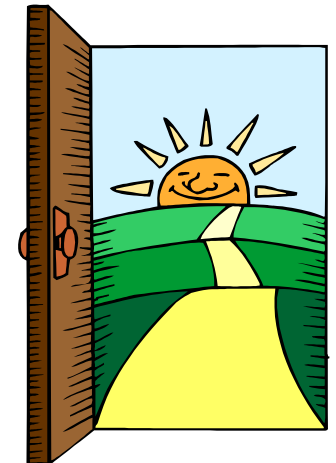


Typical and Possible Agile Practices

- 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

Workshop Discussion

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

M15

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

M17

Concurrency & 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/Agile Project Management
- **Waste in Lean/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

Workshop Discussion

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

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 Mgt.	Project Mgt.	Engr.	Engr. Support.
ML 5	OID			CAR
ML 4	OPP	QPM		
ML 3	OPF OPD OT	IPM RSKM	RD TS PI Ver Val	DAR
ML 2		PP PMC SAM	REQM	M&A PPQA CM

Process Area Valuation Approach

- **Goal Level Insufficiency** **Unacceptable**
 - Goals do not allow or support conduct of accepted lean/agile practices
- **Goal Level Sufficiency:** **Acceptable**
 - 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:** **Supportive**
 - 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:** **Enabling**
 - 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** **Unacceptable**
 - Goals do not allow or support conduct of accepted lean/agile practices
- **Goal Level Sufficiency:** **Acceptable**
 - 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:** **Supportive**
 - 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:** **Enabling**
 - 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** **Alternative**
 - Practice does not allow or support conduct of accepted lean/agile practices – Alternative practice is required
- **Supportive:** **Supportive**
 - Practice, as stated, fully supports conduct of accepted lean/agile practices
 - Informative elements are largely unhelpful
- **Enabling:** **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

Detailed project plans and oversight against project attributes presumed stable

Project Management

	PP	PMC	SAM	IPM	RSKM	IT	ISM	QPM
SG 1								
SG 2								
SG 3								
SG 4								

Engineering

	REQM	RD	TS	PI	VER	VAL
SG 1						
SG 2						
SG 3						

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

Engineering Support

	CM	PPQA	MA	DAR	OEI	CAR
SG 1						
SG 2						
SG 3						
SG 4						

Engineering support process areas are highly developed consistent with plan-based approach

Process Management

	OPF	OPD	OT	OPP	OID
SG 1					
SG 2					

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

Acceptable

SG 1 Establish Estimates

- SP 1.1 Estimate the Scope of the Project
- SP 1.2 Establish Estimates of Work
- SP 1.3 Define Project Life Cycle
- SP 1.4 Determine Estimates of Effort

Acceptable

Interpretation of phases
Iterations or hybrids of iterations-
e.g. setup
Value stream mapping is key
mechanism

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

Workflow
Options analysis
Stopping points
Exit criteria.

Close coupling of stakeholders to project activities

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

Alternative

Alternative

Enabling

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

Supportive



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

Supportive

- Waste
- Progress
- Convergence

Enabling

Enabling

- Continuous review
- Continuous waste elimination

Alternative

- Pull Kanban metrics

SG 2 Manage Corrective Action to Closure

- SP 2.1 Analyze Issues
- SP 2.2 Take Corrective Action
- SP 2.3 Manage Corrective Action

Enabling

- Agile record keeping
- Continuous corrective action

Enabling

Enabling

Enabling

Supplier Agreement Management

Enabling

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

Enabling

- Tight control of suppliers
- Fast response times

Enabling

Enabling

Enabling

- 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

Enabling

- Fast, agile

Enabling

Enabling

Enabling

Enabling

Integrated Project Management

Supportive



- 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
 - 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** Supportive Criteria

• **Learn internally and through organization** Enabling

• **Tailor very fast** Supportive
- **Extremely rapid contribution to organization's process** Enabling
- **Key to agile/lean effectiveness** Enabling

• **Close, continuously coupled coordination and collaboration** Enabling
- **Consider shared vision point architectures** Enabling
- **Consider agile team structure** Enabling

• **Critical for self-motivated teams** Enabling

● Risk Management

Enabling

SG 1 Prepare for Risk Management

Enabling

SP 1.1 Determine Risk Sources and Categories

Enabling

SP 1.2 Define Risk Parameters

Enabling

SP 1.3 Establish Risk Management Strategy

Enabling

SG 2 Identify and Analyze Risks

Enabling

SP 2.1 Identify Risks

Enabling

SP 2.2 Evaluate, Categorize, and Prioritize Risks

Enabling

SG 3 Mitigate Risks

Enabling

SP 3.1 Develop Risk Mitigation Plans

Enabling

SP 3.2 Implement Risk Mitigation Plans

Enabling

• **Good fit**
• **Ramp up to agile/lean record keeping**
• **Migrate to continuous mitigation and action**

● Integrated Teaming

Enabling

SG 1 Establish Team Composition

Enabling

SP 1.1 Identify Team Tasks

Enabling

SP 1.2 Identify Needed Knowledge and Skills

Enabling

SP 1.3 Assign Team Members

Enabling

SG 2 Govern Team Composition

Enabling

SP 2.1 Establish Team

Enabling

SP 2.2 Establish a Team Charter

Enabling

SP 2.3 Define Roles and Responsibilities

Enabling

SP 2.4 Establish Operating Procedures

Enabling

SP 2.5 Collaborate among Interfacing Teams

Enabling

• **Best CMMI support for management of tacit knowledge**
• **Overall extremely good fit for agile/lean efforts**

Integrated Supplier Management

Enabling

SG 1 Analyze and Select Sources of Products

Enabling

SP 1.1 Analyze Potential Sources of Products

Enabling

SP 1.2 Evaluate and Determine Sources of Products

Enabling

SG 2 Coordinate Work with Suppliers

Enabling

SP 2.1 Monitor Selected Supplier Processes

Enabling

SP 2.2 Evaluate Selected Supplier Work Products

Enabling

SP 2.3 Revise the Supplier Agreement or Relationship

Enabling

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.

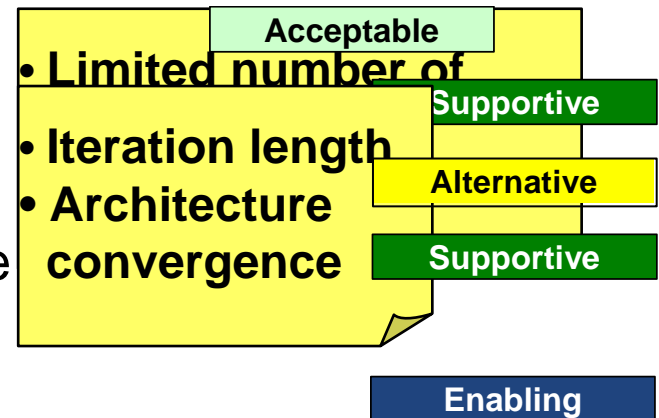
- Agile focus: reliably valuable results despite uncertainty and volatility—not predictability through invariance
- What subprocess in agile development should be “statistically managed”? Iterations? (E.g., feature points/iteration, or convergence)
- What “historical data”? From the project? From other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

Quantitative Project Management

Acceptable

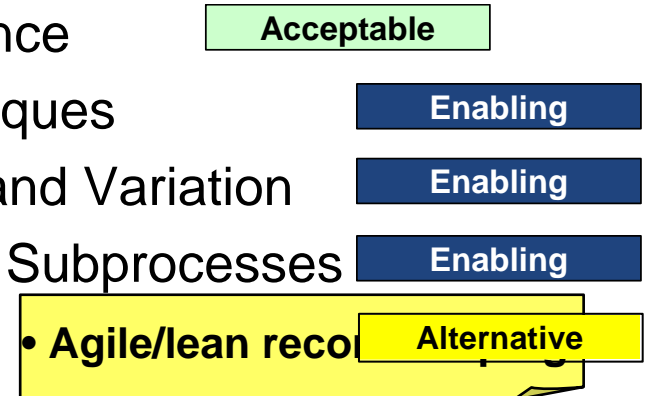
SG 1 Quantitatively Manage the Project

- 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

	PP	PMC	SAM	IPM	RSKM	IT	ISM	QPM
	Acceptable	Supportive	Enabling	Supportive	Enabling	Enabling	Enabling	Acceptable
SG 1	Acceptable	Supportive	Enabling	Supportive	Enabling	Enabling	Enabling	Acceptable
SG 2	Supportive	Enabling	Enabling	Enabling	Enabling	Enabling	Enabling	Supportive
SG 3	Acceptable			Enabling	Enabling	Enabling	Enabling	
SG 4				Enabling				

Process Area Valuation Approach

- **Goal Level Insufficiency** Unacceptable
 - Goals do not allow or support conduct of accepted lean/agile practices
- **Goal Level Sufficiency:** Acceptable
 - 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:** Supportive
 - 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:** Enabling
 - 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

- ✓ SG 1 Requirements are managed and inconsistencies with project plans and work products are identified.
 - ? SP 1.4 Maintain bidirectional traceability among the requirements and the project plans and work products.

- 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

Acceptable

SG 1 Manage Requirements

Acceptable

- 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

• “Fuzzy set” at “SW”

Supportive

Enabling

Alternative

Enabling

• Tie group design, collective code ownership, and acceptance tests to features

• Very high level traceability in lean development

Requirements Development

Supportive

SG 1	Develop Customer Requirements	Enabling	
SP 1.1	Elicit Needs		Enabling
SP 1.2	Develop the Customer Requirements		Enabling
SG 2	Develop Product Requirements		
● SP 2.1	Establish Product and Product-Component Requirements		Enabling
SP 2.2	Allocate Product-Component Requirements		Enabling
SP 2.3	Identify Interface Requirements		Enabling
SG 3	Analyze and Validate Requirements	Supportive	
SP 3.1	Establish Operational Concepts and Requirements		Enabling
● SP 3.2	Establish a Definition of Required Functionality		Supportive
SP 3.3	Analyze Requirements		Enabling
SP 3.4	Analyze Requirements to Achieve Balance		Enabling
SP 3.5	Validate Requirements with Comprehensive Methods		Enabling

• “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 a much higher level of abstraction

Technical Solution

Supportive

SG 1 Select Product-Component Solutions

Supportive

SP 1.1 Develop Detailed Alternative Solutions and Selection Criteria

SP 1.2 Evolve Operational Concepts

SP 1.3 Select Product-Component Solutions

Informative elements imply full-design-before-coding.

Supportive

Supportive

Supportive

SG 2 Develop the Design

Supportive

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

Supportive

Supportive

Supportive

Supportive

SG 3 Implement the Product Design

Supportive

SP 3.1 Implement the Design

SP 3.2 Develop Product Support Documentation

Supportive

Supportive

Product Integration

Supportive

SG 1 Prepare for Product Integration

Supportive

SP 1.1 Determine Integration Sequence

Supportive

SP 1.2 Establish the Product Integration Environment

Supportive

SP 1.3 Establish Product Integration

Supportive

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

Supportive

SG 2 Ensure Interface Compatibility

SP 2.1 Review Interface Descriptions

Supportive

SP 2.2 Manage Interfaces

Supportive

SG 3 Assemble Product Components and Deliver the Product

Supportive

SP 3.1 Confirm Readiness of Product Components for Integration

Supportive

SP 3.2 Assemble Product Components

Supportive

SP 3.3 Evaluate Assembled Product Components

Supportive

SP 3.4 Package and Deliver the Product or Product Component

Supportive

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

Acceptable



SG 1 Prepare for Verification

Enabling

SP 1.1 Select Work Products for Verification

Enabling

SP 1.2 Establish the Verification Environment

Enabling

SP 1.3 Establish Verification P

Enabling

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

SG 2 Perform Peer Reviews

SP 2.1 Prepare for Peer Review

Supportive

SP 2.2 Conduct Peer Reviews

Alternative

SP 2.3 Analyze Peer Review Data

Alternative

SG 3 Verify Selected Work Products

Acceptable

SP 3.1 Perform Verification

Supportive

SP 3.2 Analyze Verification Results and Identify Corrective Action

Alternative

Validation

Acceptable

SG 1 Prepare for Validation

Enabling

SP 1.1 Select Products for Validation

Enabling

SP 1.2 Establish the Validation Environment

Enabling

SP 1.3 Establish Validation Procedures and Criteria

Enabling

SG 2 Validate Product or Product Components

Acceptable

SP 2.1 Perform Validation

Supportive

SP 2.2 Analyze Validation Results

Alternative

Summary Valuation for Engineering

	Acceptable REQM	Supportive RD	Supportive TS	Supportive PI	Acceptable VER	Acceptable VAL
SG 1	Acceptable	Enabling	Supportive	Supportive	Enabling	Enabling
SG 2		Enabling	Supportive	Supportive	Acceptable	Acceptable
SG 3		Supportive	Supportive	Supportive	Acceptable	

Process Area Valuation Approach

- **Goal Level Insufficiency** Unacceptable
 - Goals do not allow or support conduct of accepted lean/agile practices
- **Goal Level Sufficiency:** Acceptable
 - 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:** Supportive
 - 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:** Enabling
 - 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

Configuration Management

Enabling

SG 1 Establish Baselines

Enabling

SP 1.1 Identify Configuration Items

Enabling

SP 1.2 Establish a Configuration Management

Enabling

SP 1.3 Establish Configuration Baselines

Enabling

SG 2 Track and Control Configuration Items

Enabling

SP 2.1 Identify Configuration Items

Enabling

SP 2.2 Control Configuration Items

Enabling

SG 3 Establish Integrity

Enabling

SP 3.1 Establish Configuration Management Records

Enabling

SP 3.2 Perform Configuration Audits

Enabling

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

Process and Product Quality Assurance

- SG 1 Objectively Evaluate Products
- SP 1.1 Objectively Evaluate Products
- SP 1.2 Objectively Evaluate Products and Services

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

Acceptable

Enabling

Enabling

Enabling

- SG 2 Provide Objective Input
- SP 2.1 Communicate with Noncontractors
- SP 2.2 Establish a Systematic Problem Resolution

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

Acceptable

Alternative

Alternative

Measurement and Analysis

Supportive

SG 1 Align Measurement and Analysis Activities

Supportive

SP 1.1 Establish Measurement Objectives

Enabling

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

Enabling

SP 2.1 Collect Measurement Data

Enabling

SP 2.2 Analyze Measurement Data

Enabling

SP 2.3 Store Data and Results

Enabling

SP 2.4 Communicate Results

Enabling

Decision Analysis and Resolution

Enabling

SG 1 Evaluate Alternatives

Enabling

SP 1.1 Establish Guidelines for Decision Analysis

Enabling

SP 1.2 Establish Evaluation Criteria

Enabling

SP 1.3 Identify Alternative Solutions

Enabling

SP 1.4 Select Evaluation Methods

Enabling

SP 1.5 Evaluate Alternatives

Enabling

SP 1.6 Select Solutions

Enabling

Organizational Environment for Integration

Enabling

SG 1 Evaluate Alternatives

Enabling

SP 1.1 Establish Guidelines for Decision Analysis **Enabling**

SP 1.2 Establish Evaluation Criteria **Enabling**

SP 1.3 Identify Alternative Solutions **Enabling**

SP 1.4 Select Evaluation Methods **Enabling**

SP 1.5 Evaluate Alternatives **Enabling**

SP 1.6 Select Solutions **Enabling**

Causal Analysis and Resolution

Enabling

SG 1 Determine Causes of Defects

Enabling

SP 1.1 Select Defect Data for Analysis

Enabling

SP 1.2 Analyze Causes

Enabling

SG 2 Address Causes of Defects

Enabling

SP 2.1 Implement the Action Proposals

Enabling

SP 2.2 Evaluate the Effect of Changes

Enabling

SP 2.3 Record Data

Enabling

Summary Valuation for Engineering Support

	Enabling CM	Acceptable PPQA	Supportive MA	Enabling DAR	Enabling OEI	Enabling CAR
SG 1	Enabling	Enabling	Supportive	Enabling	Enabling	Enabling
SG 2	Enabling	Acceptable	Enabling		Enabling	Enabling
SG 3	Enabling					

Process Area Valuation Approach

- **Goal Level Insufficiency** Unacceptable
 - Goals do not allow or support conduct of accepted lean/agile practices
- **Goal Level Sufficiency:** Acceptable
 - 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:** Supportive
 - 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:** Enabling
 - 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 Management

Organizational Process Focus

Supportive

- SG 1 Determine Process-Related Activities
 - SP 1.1 Establish Organizational Process Assets
 - SP 1.2 Appraise the Organizational Process Assets
 - SP 1.3 Identify the Organizational Process Assets for Improvement

By our definitions, these practices are enabling. However, informative elements to encourage agile, focused, lean and rapid process improvement are absent.

Enabling

Enabling

Enabling

Enabling

- SG 2 Plan and Implement Process-Related Activities
 - SP 2.1 Establish Process Assets
 - SP 2.2 Implement Process Assets
 - SP 2.3 Deploy Organizational Process Assets
 - SP 2.4 Incorporate Process-Related Experiences into the Organizational Process Assets

Informative elements are contrary to rapid continuous improvement

Supportive

Supportive

Enabling

Enabling

Enabling

Organizational Process Definition

Supportive

SG 1 Establish Organizational Process Assets

Supportive

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

SP 1.5

Informative elements are too focused on systemic tailoring.

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

Enabling

Organizational Training

Enabling

SG 1 Establish an Organizational Training Capability

Enabling

SP 1.1 Establish the Strategic Training Needs

Enabling

SP 1.2 Determine Which Training Needs Are the Responsibility of the Organization

Enabling

SP 1.3 Establish Organizational Training
Tactical Training

Enabling

SP 1.4 Establish Organizational Training
Tactical Training

Enabling

SG 2 Provide Necessary Training

Enabling

SP 2.1 Deliver Training

Enabling

SP 2.2 Establish Training Records

Enabling

SP 2.3 Assess Training Effectiveness

Enabling

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

Organizational Process Performance

Enabling

SG 1 Establish Performance Baselines and Models

Enabling

SP 1.1 Select Processes

Enabling

SP 1.2 Establish Process Performance Measures

Enabling

SP 1.3 Establish Quality and Process-Performance

Enabling

Ob

SP 1.4 Est Performance Baselines

Enabling

SP 1.5 Est Performance Models

Enabling

Although these are all rated as enabling, elements such as improvement of skills-based teams with highly developed tacit knowledge are missing.

Organizational Innovation and Deployment

Enabling

SG 1 Select Improvements

Enabling

SP 1.1 Collect and Analyze Improvement Proposals

Enabling

SP 1.2 Identify and Analyze Innovations

Enabling

SP 1.3 Pilot Improvements

Enabling

SP 1.4 Select Improvements for Deployment

Enabling

SG 2 Deploy Improvements

Enabling

SP 2.1 Plan the Deployment

Enabling

SP 2.2 Manage the Deployment

Enabling

SP 2.3 Measure Improvement Effects


Enabling

Summary Valuation for Process Management

	Supportive OPF	Supportive OPD	Enabling OT	Enabling OPP	Enabling OID
SG 1	Enabling	Supportive	Enabling	Enabling	Enabling
SG 2	Supportive		Enabling	Enabling	Enabling

Process Area Valuation Approach U


- **Goal Level Insufficiency** Unacceptable
 - Goals do not allow or support conduct of accepted lean/agile practices
- **Goal Level Sufficiency:** Acceptable
 - 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:** Supportive
 - 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:** Enabling
 - 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



SYSTEMS AND
SOFTWARE
CONSORTIUM
BUILDING BETTER
SOLUTIONS TOGETHER

54

©2005 Jacobs Sverdrup and
the Systems and Software Consortium, Inc.



JE JACOBS
SVERDRUP

Summary of Ratings

Project Management

	Acceptable	Supportive	Enabling	Supportive	Enabling	Enabling	Enabling	Acceptable
	PP	PMC	SAM	IPM	RSKM	IT	ISM	QPM
SG 1	Acceptable	Supportive	Enabling	Supportive	Enabling	Enabling	Enabling	Acceptable
SG 2	Supportive	Enabling	Enabling	Enabling	Enabling	Enabling	Enabling	Supportive
SG 3	Acceptable			Enabling	Enabling	Enabling	Enabling	
SG 4				Enabling				

Engineering

	Acceptable	Supportive	Supportive	Supportive	Acceptable	Acceptable
	REQM	RD	TS	PI	VER	VAL
SG 1	Acceptable	Enabling	Supportive	Supportive	Enabling	Enabling
SG 2		Enabling	Supportive	Supportive	Acceptable	Acceptable
SG 3		Supportive	Supportive	Supportive	Acceptable	

Engineering Support

	Enabling	Acceptable	Supportive	Enabling	Enabling	Enabling
	CM	PPQA	MA	DAR	OEI	CAR
SG 1	Enabling	Enabling	Supportive	Enabling	Enabling	Enabling
SG 2	Enabling	Acceptable	Enabling		Enabling	Enabling
SG 3	Enabling					

Process Management

	Supportive	Supportive	Enabling	Enabling	Enabling
	OPF	OPD	OT	OPP	OID
SG 1	Enabling	Supportive	Enabling	Enabling	Enabling
SG 2	Supportive		Enabling	Enabling	Enabling

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 2.2**
A plan for performing the process will have to be intelligently applied to avoid undue burden on agile/lean processes.
- **GP 2.5** Train the people performing or supporting the process as needed.
- **GP 2.5**
Training of agile/lean teams and team members should take advantage of
- **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 3.2 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 5.2 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

Project Management

Acceptable	Supportive	Enabling	Supportive	Enabling	Enabling	Enabling	Acceptable
PP	PMC	SAM	IPM	RSKM	IT	ISM	QPM
	Supportive	Enabling	Supportive	Enabling	Enabling	Enabling	Acceptable
		Enabling	Enabling	Enabling	Enabling	Enabling	
			Enabling	Enabling	Enabling	Enabling	
			Enabling				

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

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

Engineering

	Acceptable	Supportive	Supportive	Supportive	Acceptable	Acceptable
	REQM	RD	TS	PI	VER	VAL
SG 1	Acceptable	Enabling	Supportive	Supportive	Enabling	Enabling
SG 2		Enabling	Supportive	Supportive	Acceptable	Acceptable
SG 3		Supportive	Supportive	Supportive	Acceptable	

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

Engineering Support

	Enabling	Acceptable	Supportive	Enabling	Enabling
	CM	PPQA	MA	DAR	OEI
SG 1	Enabling	Enabling	Supportive	Enabling	Enabling
SG 2	Enabling	Acceptable	Enabling		Enabling
SG 3	Enabling				

Process Management

	Supportive	Supportive	Enabling	Enabling	Enabling
	OPF	OPD	OT	OPP	OID
SG 1	Enabling	Supportive	Enabling	Enabling	Enabling
SG 2	Supportive		Enabling	Enabling	Enabling

Workshop Discussion

Review CMMI® issues charts for closure.

References

- “Stretching Agile to fit CMMI Level 3,” David J. Anderson, http://www.agilemanagement.net/Articles/Papers/Agile_2005_Paper_DJA_v1_5.pdf
- Extreme Programming Explained: Embrace Change (2nd Edition), Ken Beck, ISBN 0-321-27865-8
- Balancing Agility and Discipline – A Guide for the Perplexed, Barry Boehm and Richard Turner, Addison Wesley
- CMMI – Guidelines for Process Integration and Product Improvement, Mary Beth Chrissis, et al, Addison Wesley
- Crystal Clear: A Human-Powered Methodology for Small Teams, Alistair Cockburn, ISBN 0-201-69947-8
- “An Agile Approach to Achieving CMMI” Christine Davis, et al, <http://www.agiletek.com/thoughtleadership/whitepapers>
- Agile/Lean CMMI Workshop – Unpublished Papers, Jeffrey L. Dutton, Jacobs Sverdrup and Rich McCabe, Systems and Software Consortium
- “Real Time Embedded Software Development Using Agile Technology,” Vincent Rivas and Joseph N Frisina, http://www.omg.org/news/meetings/workshops/RT_2005/01-4_Rivas-Frisina.pdf
- A Practical Guide to Feature-Driven Development, Stephen Palmer and John Felsing, ISBN 0-130-67615-2
- Lean Software Development – An Agile Toolkit, Mary and Tom Poppendieck, Addison Wesley
- Agile Project Management with Scrum, Ken Schwaber, ISBN 0-130-67634-9
- 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