Balancing Agility and Discipline

A Guide for the Perplexed
Two approaches to software development
- Disciplined (SW-CMM, document-based, heavy process)
- Agile (XP, tacit knowledge, light process)

Both have strengths and weaknesses
Agile and disciplined proponents are believers
Many of us are perplexed
Sources of Perplexity

• **Distinguishing method use from method misuse**
  – Claiming XP use when simply “not documenting”
  – “CMM Level 4 Memorial Library” of 99 2-inch binders

• **Overgeneralization based on the most visible instances**
  – XP is Agile; CMM is Discipline

• **Multiple definitions**
  – Quality: customer satisfaction or compliance?

• **Claims of universality**
  – The pace of IT change is accelerating and Agile methods adapt to change better than disciplined methods therefore Agile methods will take over the IT world.
  – Software development is uncertain and the SW-CMM improves predictability therefore all software developers should use the SW-CMM
Sources of Perplexity (2)

• Early success stories
  – Chrysler project that successfully birthed XP was later cancelled
  – Cleanroom has never made it into the mainstream

• Purist interpretations (and internal disagreements)
  – “Don't start by incrementally adopting parts of XP. Its pieces fit together like a fine Swiss watch”
  – "An advantage of agile methods is that you can apply them selectively and generatively."
  – "If you aren't 100% compliant with SW CMM Level 3, don't bother to bid"
  – "With the CMMI continuous interpretation, you can improve your processes in any order you feel is best."
Key Definitions

• Agile method –
  – one which fully adopts the four value propositions and twelve principles in the Agile Manifesto.

• Discipline – (per Webster)
  – control gained by enforcing obedience or order;
  – orderly or prescribed conduct or pattern of behavior;
  – self-control.

• Plan-driven –
  – a description for disciplined methods (order is often defined in plans)

• Plan – (per Webster)
  – a method for achieving an end (a process plan);
  – an orderly arrangements of parts of an overall design (a product plan).
  – We assume that such plans are documented.
Finding Middle Ground

- A pragmatic view
- Both approaches have “home grounds”
- A broad range of environments and needs
- Trends require better handling of rapid change
- Processes should be the right weight for the job
- We believe risk-based analysis is the key to finding middle ground
Contrasts and Home Grounds

• Comparison is difficult, but we found 4 areas where there are clear differences

• Application
  – The type of project

• Management
  – Communication and control aspects

• Technical
  – How the software is developed

• Personnel
  – The type and competency of developers and stakeholders
Application Characteristics

• Primary goals
  – A: Rapid value, responsiveness to change
  – D: Predictability, stability, high assurance

• Size
  – A: Small to medium
  – D: Large

• Environment
  – A: Turbulent, high change
  – D: Stable, requirements defined
Management Characteristics

• Customer Relations
  – A: Dedicated, collocated
  – D: Contractual
  – A: Trust through working software and participation
  – D: Trust through process maturity evaluations

• Planning and Control
  – A: Means to an end
  – D: Anchor processes, communication

• Project Communications
  – A: Tacit, interpersonal knowledge
  – D: Explicit, documented knowledge
Technical Characteristics

• Requirements
  – A: Adjustable, informal stories
  – D: Formally baselined, complete, consistent specifications

• Development
  – A: Simple Design
  – D: Architecture-based design

• Testing
  – A: Automated, test-driven
  – D: Planned, requirements-driven
Technical Characteristics (2)

Cost of Change

Beck

Project 1- Activities Growth

Li

Project 2- Activities Growth

Li
Technical Characteristics (3)
How Much Architecting?

Sweet Spot Drivers:
Rapid Change: leftward
High Assurance: rightward
Personnel Characteristics

• Customers
  – A: CRACK customers throughout development
  – D: CRACK customers early
    • CRACK: Collaborative, Representative, Authorized, Committed, and Knowledgeable

• Developers
  – A: Heavy mix of high caliber throughout
  – D: Heavy mix early with lower capability later

• Culture
  – A: Many degrees of freedom (Thrives on chaos)
  – D: Clear policies and procedures (Thrives on order)
## Personnel Characteristics (2)

### Modified Cockburn Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Able to revise a method (break its rules) to fit an unprecedented new situation</td>
</tr>
<tr>
<td>2</td>
<td>Able to tailor a method to fit a precededented new situation</td>
</tr>
<tr>
<td>1A</td>
<td>With training, able to perform discretionary method steps (e.g., sizing stories to fit increments, composing patterns, compound refactoring, complex COTS integration). With experience can become Level 2.</td>
</tr>
<tr>
<td>1B</td>
<td>With training, able to perform procedural method steps (e.g. coding a simple method, simple refactoring, following coding standards and CM procedures, running tests). With experience can master some Level 1A skills.</td>
</tr>
<tr>
<td>-1</td>
<td>May have technical skills, but unable or unwilling to collaborate or follow shared methods.</td>
</tr>
</tbody>
</table>

![Graph showing the relationship between Adaptation and Rigor]
Summary of Home Grounds

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Agile</th>
<th>Disciplined</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Primary Goals</td>
<td>Rapid value; responding to change</td>
<td>Predictability, stability, high assurance</td>
</tr>
<tr>
<td>Size</td>
<td>Smaller teams and projects</td>
<td>Larger teams and projects</td>
</tr>
<tr>
<td>Environment</td>
<td>Turbulent; high change; project-focused</td>
<td>Stable; low-change; project/organization focused</td>
</tr>
<tr>
<td><strong>Management</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer Relations</td>
<td>Dedicated on-site customers; focused on prioritized increments</td>
<td>As-needed customer interactions; focused on contract provisions</td>
</tr>
<tr>
<td>Planning/Control</td>
<td>Internalized plans; qualitative control</td>
<td>Documented plans, quantitative control</td>
</tr>
<tr>
<td>Communications</td>
<td>Tacit interpersonal knowledge</td>
<td>Explicit documented knowledge</td>
</tr>
<tr>
<td><strong>Technical</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Requirements</td>
<td>Prioritized informal stories and test cases; undergoing unforeseeable change</td>
<td>Formalized project, capability, interface, quality, foreseeable evolution requirements</td>
</tr>
<tr>
<td>Development</td>
<td>Simple design; short increment; refactoring assumed inexpensive</td>
<td>Extensive design; longer increments; refactoring assumed expensive</td>
</tr>
<tr>
<td>Test</td>
<td>Executable test cases define requirements, testing</td>
<td>Documented test plans and procedures</td>
</tr>
<tr>
<td><strong>Personnel</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customers</td>
<td>Dedicated, collocated CRACK* performers</td>
<td>CRACK* performers, not always collocated</td>
</tr>
<tr>
<td>Developers</td>
<td>At least 30% full-time Cockburn level 2 and 3 experts; no Level 1B or -1 personnel**</td>
<td>50% Cockburn Level 3s early; 10% throughout; 30% Level 1B’s workable; no Level -1s**</td>
</tr>
<tr>
<td>Culture</td>
<td>Comfort and empowerment via many degrees of freedom (thriving on chaos)</td>
<td>Comfort and empowerment via framework of policies and procedures (thriving on order)</td>
</tr>
</tbody>
</table>

* Collaborative, Representative, Authorized, Committed, Knowledgeable
** These numbers will particularly vary with the complexity of the application
Five Critical Decision Factors

- Represent five dimensions
- Size, Criticality, Dynamism, Personnel, Culture
Two Case Studies

- Show how that agile and disciplined methods can be extended
- Provide examples of success and lessons learned
- ThoughtWorks Lease Management
  - Agile extended with disciplined
- CCPDS-R
  - Disciplined overlaid with agile concepts
Thoughtworks Lease Management

- XP replaced ineffective traditional development
- Problems when project moved beyond XP assumptions
  - The effort to develop or modify a story really does not increase with time and story number
  - Trusting people to get everything done on time is compatible with fixed schedules and diseconomies of scale
  - Simple design and YAGNI scale up easily to large projects
- Disciplined practices enabled XP to scale up
  - High-level architectural plans to provide essential big-picture information
  - More careful definition of milestone completion criteria to avoid “finishing” but not finishing
  - Use of design patterns and architectural solutions rather than simple design to handle foreseeable change
CCPDS-R

- USAF Command Center Processing and Display System Replacement for early missile warning
- Government contract environment required disciplined approach, project needed agility
- Practices implemented to provide agility mapped well to the Agile Manifesto
  - Individuals and Interactions over Processes and Tools
    - Milestone content was redefined
    - Architecture was organized around the performers’ skill levels
  - Working Software over Comprehensive Documentation
    - Later PDR demonstrated working software for high-risk areas
  - Customer Collaboration Over Contract Negotiation
    - Used COCOMO for cost and schedule negotiations
  - Responding to Change Over Following a Plan
CCPDS-R (2) Cost of Change

Project Development Schedule

Design Changes
Implementation Changes
Maintenance Changes and ECP’s

Cost of Change

Beck

Time
Using Risk to Balance Discipline and Agility - Overview

Step 1. Risk Analysis
- Rate the project's environmental, agility-oriented and discipline-oriented risks.
- Uncertain about ratings?
  - Yes: Buy information via prototyping, data collection and analysis
  - No: Go Risk-driven

Step 2. Risk Comparison
- Compare the agile and disciplined risks
  - Agility risks dominate
    - Go Risk-driven Agile
  - Discipline risks dominate
    - Go Risk-driven Disciplined
  - Neither dominate
    - Go Risk-driven Agile in agile parts; Go Risk-driven Disciplined elsewhere

Step 3. Architecture Analysis
- Architect application to encapsulate agile parts

Step 4. Tailor Life Cycle
- Tailor life cycle process around anchor point commitment milestones

Step 5. Execute and Monitor
- Deliver incremental capabilities according to strategy
- Monitor progress and risks/opportunities, readjust balance and process as appropriate
Risks Used in Risk Comparison

- **Environmental risks**
  - Technology uncertainties
  - Many stakeholders
  - Complex system-of-systems

- **Agility-oriented risks**
  - Scalability
  - Use of simple design
  - Personnel turnover
  - Too-frequent releases
  - Not enough agile-capable people

- **Discipline-oriented risks**
  - Rapid change
  - Need for rapid results
  - Emergent requirements
  - Not enough discipline-capable people
Three Examples

• Agent-based systems
• Small – Event Managers application
• Medium – SupplyChain.com application
• Large – National Information System for Crisis Management (NISCM) application
• Each example results in a development strategy based on risk analyses
Event Managers Project

• Small startup company
• Diverse set of smaller agent-based planning applications
• This project: support for managing the infrastructure and operations of conferences and conventions
• Widening variety of options and interdependencies, makes an agent-based approach highly attractive
# Event Managers Profile

## Risk Items

<table>
<thead>
<tr>
<th>Risk Items</th>
<th>Risk Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Environmental Risks</strong></td>
<td>Event Managers</td>
</tr>
<tr>
<td>E1. Technology uncertainties</td>
<td>[ ]</td>
</tr>
<tr>
<td>E2. Many stakeholders</td>
<td>[ ]</td>
</tr>
<tr>
<td>E3. Complex system of systems</td>
<td>[ ]</td>
</tr>
<tr>
<td><strong>Risks of using Agile Methods</strong></td>
<td></td>
</tr>
<tr>
<td>A1. Scalability</td>
<td>[ ]</td>
</tr>
<tr>
<td>A2. Use of simple design</td>
<td>[ ]</td>
</tr>
<tr>
<td>A3. Personnel turnover</td>
<td>[ ] and [ ]</td>
</tr>
<tr>
<td>A4. Too-frequent releases</td>
<td>[ ]</td>
</tr>
<tr>
<td>A5. Not enough agile-capable people</td>
<td>[ ]</td>
</tr>
<tr>
<td><strong>Risks of using disciplined methods</strong></td>
<td></td>
</tr>
<tr>
<td>D1. Rapid change</td>
<td>[ ] and [ ]</td>
</tr>
<tr>
<td>D2. Need for rapid results</td>
<td>[ ] and [ ]</td>
</tr>
<tr>
<td>D3. Emergent requirements</td>
<td>[ ] and [ ]</td>
</tr>
<tr>
<td>D4. Not enough discipline-capable people</td>
<td>[ ]</td>
</tr>
</tbody>
</table>

## Risk rating scale:

- [ ] - minimal risk
- [ ] - moderate risk
- [ ] - Serious but manageable risk
- [ ] - Very serious but manageable risk
- [ ] - Show-stopper risk

## Diagrams

- **Personnel** (% Level 1B) (% Level 2&3)
- **Criticality** (Loss due to impact of defects)
- **Dynamism** (% Requirements-change/month)
- **Size** (# of personnel)
- **Culture** (% thriving on chaos vs. order)
1. Startup
Teambuilding and Shared Vision

- Establish CRACK joint management team
- Develop shared vision, results chain, life cycle strategy, infrastructure choices
- Establish commitment to proceed, incremental capabilities, backlog

2. Design, Development and Deployment

- Test, exercise, deploy new increment
- Re-prioritize next-increment features
- Analyze lessons learned; prepare strategy for next increment

Customer Management; Representative Users

Joint Developer-Customer Agile Team
SupplyChain.com Profile

- Turnkey agent-based supply chain management systems
- Distributed, multi-organization teams of about 50 people
- Parts of applications are relatively stable, while others are highly volatile
- Architectures are driven by a few key COTS packages that are also continually evolving
### SupplyChain.com Profile

<table>
<thead>
<tr>
<th>Risk Items</th>
<th>Risk Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Environmental Risks</strong></td>
<td></td>
</tr>
<tr>
<td>E1. Technology uncertainties</td>
<td>★★</td>
</tr>
<tr>
<td>E2. Many stakeholders</td>
<td>★★</td>
</tr>
<tr>
<td>E3. Complex system of systems</td>
<td>★★</td>
</tr>
<tr>
<td><strong>Risks of using Agile Methods</strong></td>
<td></td>
</tr>
<tr>
<td>A1. Scalability</td>
<td>★★</td>
</tr>
<tr>
<td>A2. Use of simple design</td>
<td>★★</td>
</tr>
<tr>
<td>A3. Personnel turnover</td>
<td>★★</td>
</tr>
<tr>
<td>A4. Too-frequent releases</td>
<td>★★</td>
</tr>
<tr>
<td>A5. Not enough agile-capable people</td>
<td>★★</td>
</tr>
<tr>
<td><strong>Risks of using disciplined methods</strong></td>
<td></td>
</tr>
<tr>
<td>D1. Rapid change</td>
<td>★★</td>
</tr>
<tr>
<td>D2. Need for rapid results</td>
<td>★★</td>
</tr>
<tr>
<td>D3. Emergent requirements</td>
<td>★★</td>
</tr>
<tr>
<td>D4. Not enough discipline-capable people</td>
<td>★★</td>
</tr>
</tbody>
</table>

**Risk rating scale:**
- ★ - minimal risk
- ★★★ - moderate risk
- ★★★★ - Serious but manageable risk
- ★★★★★ - Very serious but manageable risk
- ★★★★★★ - Show-stopper risk

### Risk Items

- **Environmental Risks**
  - E1. Technology uncertainties
  - E2. Many stakeholders
  - E3. Complex system of systems

- **Risks of using Agile Methods**
  - A1. Scalability
  - A2. Use of simple design
  - A3. Personnel turnover
  - A4. Too-frequent releases
  - A5. Not enough agile-capable people

- **Risks of using disciplined methods**
  - D1. Rapid change
  - D2. Need for rapid results
  - D3. Emergent requirements
  - D4. Not enough discipline-capable people

**Risk rating scale:**
- ★ - minimal risk
- ★★★ - moderate risk
- ★★★★ - Serious but manageable risk
- ★★★★★ - Very serious but manageable risk
- ★★★★★★ - Show-stopper risk

### Diagram

- **Personnel** (% Level 1B) (% Level 2&3)
- **Criticality** (Loss due to impact of defects)
- **Dynamism** (% Requirements-change/month)
- **Size** (% of personnel)
- **Culture** (% thriving on chaos vs. order)
SupplyChain.com Strategy

1. Teambuilding and Shared Vision
   - Furnish CRACK representatives and alternates
   - Develop benefits-realization results chains
   - Identify missing stakeholders
   - Enhance mutual understanding
   - Explore goals, options, technology, prototypes, risks

2. Systems Definition and Architecting
   - Elaborate supply chain operational concept, prototypes, transition strategy
   - Evaluate and determine COTS, reuse, and architecture choices
   - Prioritize and sequence desired capabilities, outstanding risks
   - Establish project organization, overall process, and feature teams

3. Design, Development and Deployment
   - Ensure representative exercise of incremental capabilities
   - Monitor progress and new operational development
   - Communicate proposed redirections
   - Prepare for and execute acceptance tests, installation, operational evolution
   - Use risk to determine content of artifacts
   - Monitor project progress, risk resolution, and new technology developments
   - Identify and work critical project-level actions
   - Develop and integrate new feature increments
   - Analyze feedback on current version
   - Reprioritize features
   - Prepare strategy for next increment
   - Consolidate process lessons learned
NISCM Profile

- Broad coalition of government agencies and critical private-sector organizations
- Support cross-agency and public-private sector coordination of crisis management activities
- Adaptive mobile network, virtual collaboration, information assurance, and information integration technology
- Private-sector system-of-systems integration contractor
<table>
<thead>
<tr>
<th>Risk Items</th>
<th>Risk Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NISCM</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Environmental Risks</strong></td>
<td></td>
</tr>
<tr>
<td>E1. Technology uncertainties</td>
<td>■■■</td>
</tr>
<tr>
<td>E2. Many stakeholders</td>
<td>■■■</td>
</tr>
<tr>
<td>E3. Complex system of systems</td>
<td>■■■</td>
</tr>
<tr>
<td><strong>Risks of using Agile Methods</strong></td>
<td></td>
</tr>
<tr>
<td>A1. Scalability</td>
<td>■■ — ■■■■■</td>
</tr>
<tr>
<td>A2. Use of simple design</td>
<td>■■ — ■■■■■</td>
</tr>
<tr>
<td>A3. Personnel turnover</td>
<td>■■</td>
</tr>
<tr>
<td>A4. Too-frequent releases</td>
<td>■■</td>
</tr>
<tr>
<td>A5. Not enough agile-capable people</td>
<td>■■ — ■■■■■</td>
</tr>
<tr>
<td><strong>Risks of using disciplined methods</strong></td>
<td></td>
</tr>
<tr>
<td>D1. Rapid change</td>
<td>■■</td>
</tr>
<tr>
<td>D2. Need for rapid results</td>
<td>■■</td>
</tr>
<tr>
<td>D3. Emergent requirements</td>
<td>■■</td>
</tr>
<tr>
<td>D4. Not enough discipline-capable people</td>
<td>■■</td>
</tr>
</tbody>
</table>

**Risk rating scale:**
- : - minimal risk
- ■ - moderate risk
- ■■ - serious but manageable risk
- ■■■ - very serious but manageable risk
- ■■■■ - show-stopper risk

![Diagram](image-url)
NISCM Strategy

**Stakeholders**
- NSO and I3-led Project Risk Management Team

**Startup**
- Furnish CRACK representatives and alternates
- Staff and organize to cover all success-critical risk areas
- Prepare for and select competitors for concept definition

**Teambuilding and Shared Vision**
- Develop results chains
  - Identify missing stakeholders
  - Enhance understanding
  - Explore goals, options, technology, prototypes, risks
  - Negotiate mutually satisfactory objectives and priorities
- Formulate top-level operational concept, requirements, architecture, life cycle plan, feasibility rational
- Select winning system integration contract

**Systems Definition and Architecting**
- Prepare for/select competitors for component subcontract definition
- Elaborate operational concept, prototypes, transition strategy
- Evaluate/determine COTS, reuse, and architecture choices
- Develop/validate critical-infrastructure software
- Prioritize/sequence desired capabilities, outstanding risks
- Formulate definitive operational concept, requirements, architecture, life cycle plan, feasibility rationale
- Use to solicit/select component subcontractors

**Hold Life Cycle Objective Review. If feasibility rationale is valid, commit to proceed**

**Design, Development and Deployment**
- Ensure representative exercise of incremental capabilities
- Monitor progress and new operational development
- Use risk to determine content of artifacts
- Communicate proposed redirections

**Hold Life Cycle Architecture Review. If feasibility rationale is valid, commit to proceed**

**Stable, Higher-criticality Module Teams**
- Develop compatible component-level prototypes, requirements, architectures, plans, critical software, feasibility rationale
- Classify modules by likely stability and criticality

**Dynamic, Lower-criticality Module Teams**
- Communicate proposed redirections
- Prepare for and execute acceptance tests, installation, operational evolution
- Plan, specify, develop dynamic, lower-criticality module increments

- Analyze feedback on current version
- Reprioritize features
- Prepare strategy for next increment

- Plan, specify, develop stable, higher-criticality module increments
## Risk Profiles Across Examples

<table>
<thead>
<tr>
<th>Risk Items</th>
<th>Risk Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Event Managers</td>
</tr>
<tr>
<td><strong>Environmental Risks</strong></td>
<td></td>
</tr>
<tr>
<td>E1. Technology uncertainties</td>
<td>☐</td>
</tr>
<tr>
<td>E2. Many stakeholders</td>
<td>☐</td>
</tr>
<tr>
<td>E3. Complex system of systems</td>
<td>☐</td>
</tr>
<tr>
<td><strong>Risks of using Agile Methods</strong></td>
<td></td>
</tr>
<tr>
<td>A1. Scalability</td>
<td>☐</td>
</tr>
<tr>
<td>A2. Use of simple design</td>
<td>☐</td>
</tr>
<tr>
<td>A3. Personnel turnover</td>
<td>☐</td>
</tr>
<tr>
<td>A4. Too-frequent releases</td>
<td>☐</td>
</tr>
<tr>
<td>A5. Not enough agile-capable people</td>
<td>☐</td>
</tr>
<tr>
<td><strong>Risks of using disciplined methods</strong></td>
<td></td>
</tr>
<tr>
<td>D1. Rapid change</td>
<td>☐</td>
</tr>
<tr>
<td>D2. Need for rapid results</td>
<td>☐</td>
</tr>
<tr>
<td>D3. Emergent requirements</td>
<td>☐</td>
</tr>
<tr>
<td>D4. Not enough discipline-capable people</td>
<td>☐</td>
</tr>
<tr>
<td><strong>Risk rating scale:</strong></td>
<td></td>
</tr>
<tr>
<td>☐ - minimal risk</td>
<td>☐</td>
</tr>
<tr>
<td>■ - moderate risk</td>
<td>☐</td>
</tr>
<tr>
<td>■■ - Serious but manageable risk</td>
<td></td>
</tr>
<tr>
<td>■■■■ - Very serious but manageable risk</td>
<td></td>
</tr>
<tr>
<td>■■■■■ - Show-stopper risk</td>
<td></td>
</tr>
</tbody>
</table>
Conclusions

• Neither agile nor disciplined methods provide a silver bullet
• Agile and disciplined methods have home grounds where one clearly dominates the other
• Future trends are toward application developments that need both agility and discipline
• Some balanced methods are emerging
• It is better to build your method up than to tailor it down
• Methods are important, but potential silver bullets are more likely to be found in areas dealing with people, values, communications, and expectations management.