Disciplined Agile Delivery (DAD) in a Nutshell

Mark Lines
mark@scottwambler.com
twitter.com/mark_lines
Book cover pictures that didn’t make it...
What I am going to cover in my talk...

• Understand what DAD is and why we need it

• Discover why DAD is called a “process decision framework”

• Understand the basic and advanced DAD Lifecycles

• Learn how DAD is goal-driven

• To be introduced to the three phases of the DAD lifecycle
Agenda

- Disciplined Agile Delivery (DAD)
- Characteristics of Good Teams
- A Hybrid Framework
- Potential DAD Lifecycles
- Comparing Terminology
- Enterprise Awareness
- Goal-Driven, Not Prescriptive
- How it Works in Practice
- Tailoring and Scaling Agile
Disciplined Agile Delivery (DAD)

Disciplined Agile Delivery (DAD) is a process decision framework

The key characteristics of DAD:

- People-first
- Goal-driven
- Hybrid agile
- Learning-oriented
- Full delivery lifecycle
- Solution focused
- Risk-value lifecycle
- Enterprise aware
Characteristics of Good Teams

• The majority of team members should be “generalizing specialists”
  – Also known as “T-Skilled” people

• DAD teams and team members should be:
  – Self-disciplined in that they commit only to the work which they can accomplish and then perform that work as effectively as possible.
  – Self-organizing, in that they will estimate and plan their own work and then proceed to collaborate iteratively to do so.
  – Self-aware, in that they strive to identify what works well for them, what doesn’t, and then learn and adjust accordingly.
DAD is a Hybrid Framework

DAD leverages proven strategies from several sources, providing a decision framework to guide your adoption and tailoring of them in a context-driven manner.

© Scott W. Ambler + Associates
### Agile Sources for DAD

<table>
<thead>
<tr>
<th>Agile Source</th>
<th>Strengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scrum</td>
<td>Project management framework; release, iteration and daily planning; prioritization/scope management; regular stakeholder reviews; retrospectives to help the team evolve; cross-functional team of generalizing specialists</td>
</tr>
<tr>
<td>Extreme Programming (XP)</td>
<td>Technical aspects of software development with specific practices defined for fine-scale feedback, continuous integration, shared understanding, and programmer welfare</td>
</tr>
<tr>
<td>Agile Modeling</td>
<td>Lightweight requirements, architecture, and design modeling and documentation</td>
</tr>
<tr>
<td>Agile Data</td>
<td>Database architecture, design, and development</td>
</tr>
<tr>
<td>Kanban and Lean Software Development</td>
<td>A collection of principles and strategies that help streamline software development and provide advice for scaling agile approaches</td>
</tr>
<tr>
<td>Unified Process (UP)</td>
<td>Full delivery lifecycle planning, modeling, development, testing, deployment, and governance</td>
</tr>
<tr>
<td>Other</td>
<td>Various ideas and techniques have also been adopted from Crystal, Dynamic System Development Method (DSDM), Outside In Development (OID), Feature Driven Development (FDD), and the IBM Practices Library</td>
</tr>
</tbody>
</table>
DAD Lifecycle: Basic/Agile

Inception
- One or more short iterations
- Stakeholder consensus
- Proven architecture

Construction
- Many short iterations producing a potentially consumable solution each iteration
- Project viability (several)
- Sufficient functionality

Transition
- One or more short iterations
- Production ready
- Delighted stakeholders

© Scott W. Ambler + Associates
DAD Lifecycle: Advanced/Lean

[Diagram showing the lifecycle stages and processes, including Inception, Construction, and Transition phases with various activities and feedback loops.]
The Phases Disappear Over Time

First release: Inception Construction Transition

Second release: I Construction T

Third release: I Construction T

Nth+ releases: C T C T C T
Enterprise Awareness

• Consider how your agile teams need to be “Enterprise Aware”:
  – What other teams might an agile team need to interact with in your organization?
  – Do these teams work in an agile manner? If not, what are you doing to address this?
  – What information do your agile teams need to provide to senior management for governance purposes? Why?
  – Are your agile teams expected to conform to an existing technical architecture? Organizational business vision? If so, how is this supported?
  – Do you have coding guidelines to follow? Data guidelines? Usability? Security? Other? How are they supported or enforced?

• Mainstream agile methods sometimes assume that each project is a small, self-sufficient team
  – We know that this seldom a reality of complex enterprise projects
  – Scrum stresses inward focus of the team with minimal distractions
  – DAD adds to this by addressing the necessity to work outside the team with other projects and stakeholders
## Comparing DAD and Scrum Terminology

<table>
<thead>
<tr>
<th>DAD Term</th>
<th>Scrum Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration</td>
<td>Sprint</td>
</tr>
<tr>
<td>Team lead</td>
<td>ScrumMaster*</td>
</tr>
<tr>
<td>Coordination meeting</td>
<td>(Daily) Scrum meeting</td>
</tr>
<tr>
<td>Retrospective</td>
<td>Sprint retrospective</td>
</tr>
<tr>
<td>Demo</td>
<td>Sprint demo</td>
</tr>
</tbody>
</table>

* These roles aren’t completely the same, but close
## DAD is Goal-Driven

<table>
<thead>
<tr>
<th>Goals for the Inception Phase</th>
<th>Goals for Construction Phase Iterations</th>
<th>Goals for the Transition Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Form initial team</td>
<td>- Produce a potentially consumable solution</td>
<td>- Ensure the solution is consumable</td>
</tr>
<tr>
<td>- Develop common project vision</td>
<td>- Address changing stakeholder needs</td>
<td>- Deploy the solution</td>
</tr>
<tr>
<td>- Align with enterprise direction</td>
<td>- Move closer to deployable release</td>
<td></td>
</tr>
<tr>
<td>- Explore initial scope</td>
<td>- Improve quality</td>
<td></td>
</tr>
<tr>
<td>- Identify initial technical strategy</td>
<td>- Prove architecture early</td>
<td></td>
</tr>
<tr>
<td>- Develop initial release plan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Form work environment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Secure funding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Identify risks</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Ongoing Goals

- Fulfill the project mission
- Grow team members
- Address risk

- Improve team process and environment
- Leverage and enhance existing infrastructure
Goal: Develop Common Vision

- Vision Strategy
  - Stakeholder driven
  - Team driven
  - Sponsor driven
  - Collaborative

- Level of Detail
  - Information radiators
  - Light-weight
  - Detailed
  - None

- Level of Agreement
  - Consensus
  - General agreement
  - Dictated
  - None
### The Agile 3C (Coordinate-Collaborate-Conclude) Rhythm

<table>
<thead>
<tr>
<th>Release rhythm</th>
<th>Inception</th>
<th>Construction</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day to weeks</td>
<td>Several iterations</td>
<td>Hours to weeks</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Iteration rhythm</th>
<th>Iteration planning</th>
<th>Development</th>
<th>Iteration wrap up</th>
</tr>
</thead>
<tbody>
<tr>
<td>A few hours</td>
<td>Several days</td>
<td>A few hours</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Daily rhythm</th>
<th>Coordination Meeting</th>
<th>Daily Work</th>
<th>Stabilize</th>
</tr>
</thead>
<tbody>
<tr>
<td>A few minutes</td>
<td>Several hours</td>
<td>Varies</td>
<td></td>
</tr>
</tbody>
</table>

**Coordinate**

**Collaborate**

**Conclude**

© Scott W. Ambler + Associates
The Inception phase

- Initiate team
- Schedule stakeholders for envisioning sessions

Coordinate
- Up to a few hours

Project Selected

Collaborate
- Ideally: Up to a few weeks
  - Average: 4 weeks
  - Worst case: Several months

Conclude
- Up to a few hours
- Light-weight milestone review
- Communicate vision to stakeholders

Stakeholder consensus
The Construction phase

- **Coordinate**
  - Prove the architecture works via end-to-end working slice of the solution
  - Typical: 1 iteration
  - Worst case: Many iterations

- **Collaborate**
  - Incrementally produce a consumable solution
  - Share project status with stakeholders
  - Align with organizational goals
  - Align with other project teams
  - Improve individual and team performance
  - Several iterations

- **Conclude**
  - Determine sufficiency
  - Harden the solution
  - Ideally: Several hours
  - Sufficient Functionality
A Construction Iteration

**Coordinate**
- Iteration planning
- Iteration modeling

**Collaborate**
- "Standard" practices:
  - Visualize work
  - Daily coordination meeting
  - Refactoring
  - Developer regression testing
  - Model storming
  - Continuous integration (CI)
  - Sustainable pace
  - Prioritized requirements
  - Architecture spike
  - Collective ownership
  - Burn-down chart
  - Automated metrics

- "Advanced" practices:
  - Test-driven development (TDD)
  - Acceptance TDD (ATDD)
  - Continuous deployment (CD)
  - Look-ahead modeling
  - Parallel independent testing
  - Continuous documentation
  - Non-solo development
  - Look-ahead planning

**Conclude**
- Iteration demo
- Retrospective
- Release planning (update)
- Determine "go forward" strategy

2 hours for each week of the iteration length

Typical: One to four weeks
Average: Two weeks
Worst case: Six weeks

One hour per week of iteration length

Potentially consumable solution
A Typical Day of Construction

Coordinate
- Daily coordination meeting
- Update task board
- Update iteration burndown

Collaborate
- Address blocking issues
- Create tests
- Develop code
- Integrate
- Fix problems
- Model storm
- Deploy to test/demo environment

Conclude
- Stabilize build

Start of day
Up to 15 minutes
Typical: 5-6 hours
Ideally: Not a concern
End of day
The Transition phase

- **Coordinate**
  - Phase planning
  - Transition planning
  - End-of-lifecycle testing and fixing
  - Data and user migration
  - Pilot/beta the solution
  - Finalize documentation
  - Communicate deployment
  - Prepare support environment
  - Train/educate stakeholders

- **Collaborate**
  - Ideally: Nothing
  - Typical: One hour per week of collaborate time
  - Average: 4 weeks
  - Worst case: Several months

- **Conclude**
  - Production readiness review
  - Deploy solution
  - Ideally: Less than an hour
  - Worst case: Several months
  - Production Ready

**Sufficient Functionality**

**Delighted Stakeholders**
Context Counts – Tailoring and Scaling Agile

Disciplined agile delivery with one or more complexity factors:

- Large teams
- Geographically distributed teams
- Compliance
- Domain or technical complexity
- Cultural/organizational issues
- Organizational distribution

Agility at Scale

- Delivery focus
- Risk-value driven lifecycle
- Self-organization with appropriate governance
- Goal driven
- Enterprise aware

Disciplined Agile Delivery

- Construction focus
- Value driven lifecycle
- Self-organizing teams
- Prescriptive
- Project team aware

Agile
Summary

- DAD adds value to existing mainstream agile methods in these ways:
  - Full lifecycle coverage of practices
  - Recognition of project phases and lightweight milestones
  - Removal of proprietary terminology
  - Addresses enterprise concerns such as governance, enterprise authorities
  - Foundation for scaling agile beyond small co-located teams
For more information...

- The “Disciplined Agile Delivery Experience”
  - 3-day workshop, open enrolment or on-site upon request
  - Emphasis is on learning, not slideware
  - Attendees receive a copy of Mark and Scott’s book, and decks of Planning Poker estimating cards

- The DAD community website
  - www.DisciplinedAgileDelivery.com
Thank You!

Mark@scottwambler.com
Twitter: mark_lines

AgileModeling.com
AgileData.org
Ambysoft.com
DisciplinedAgileDelivery.com
EnterpriseUnifiedProcess.com
ScottWAmbler.com
Recommended Resources

[Images of various books related to agile delivery, agile modeling, data, and processes]