Disciplined Agile Delivery - A Tutorial

Mark Lines
mark@scottambler.com
Mark_Lines

Objectives of this Workshop

- The book is 500 pages
- We have 3 hours
- My assumption is that you have basic knowledge of mainstream agile methods such as Scrum
  - Valid assumption?
- So let’s focus on
  - What is Disciplined Agile Delivery (DAD)?
  - Why do we need it?
  - How does DAD compare mainstream agile methods?
  - How to learn more
Introductions

• A big group so we won’t do individual introductions
• Does anyone have something specific that they want to learn about Disciplined Agile Delivery today?

Criteria for a Disciplined Agile Team

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business value</td>
<td>Produce a working solution on a regular basis which provides quantifiable value to stakeholders</td>
</tr>
<tr>
<td>Validation</td>
<td>Do continuous regression testing, and better yet take a Test-Driven Development (TDD) approach</td>
</tr>
<tr>
<td>Collaboration</td>
<td>Work closely with their stakeholders, or a stakeholder proxy, ideally on a daily basis</td>
</tr>
<tr>
<td>Self organization</td>
<td>Are self-organizing and work within an appropriate governance framework</td>
</tr>
<tr>
<td>Improvement</td>
<td>Regularly reflect on, and measure, how they work together and then act to improve on their findings in a timely manner</td>
</tr>
</tbody>
</table>
Exercise: How Agile Are You?

• Get back into your teams

• For 5 minutes, rate yourself and your organization on a scale of one to five for the following criteria:
  — Business value (1 = Infrequent releases, 3 = Quarterly Releases, 5 = Weekly or better releases)
  — Validation (1 = Separate testers do the testing, 3 = Developers do some regression testing, 5 = test driven approach)
  — Collaboration (1 = We mostly see stakeholders at beginning and end of project, 3 = We have weekly stakeholder contact, 5 = Daily stakeholder contact)
  — Self Organization (1 = Management plans, 3 = The team plans, 5 = The team plans and is governed in an agile manner)
  — Improvement (1 = We have a project post mortem, 3 = We have regular retrospectives, 5 = We have retrospectives and measure our progress)

Agenda

• A Disciplined Agile Manifesto
• An Overview of Disciplined Agile Delivery (DAD)
• DAD Roles
• The Inception Phase
• The Construction Phase
• The Transition Phase
• Governing Disciplined Agile Teams
• Comparing the DAD and SAFe Frameworks
The Agile Manifesto

We value:

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

That is, while there is value in the items on the right, we value the items on the left more.

Source: www.agilemanifesto.org

But... DAD Extends Agile Thinking

- Solutions, not just software
- Stakeholders, not just customers
- The organizational ecosystem, not just development teams
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
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.

Full Delivery Lifecycle: A High-Level View

DAD is NOT Water-scrum-fall!!
Disciplined Agile Delivery Tutorial

DAD Lifecycle: Basic/Agile

DAD Lifecycle: Advanced/Lean
**The Phases Disappear Over Time**

First release:  

| Inception | Construction | Transition |

Second release:  

| I | Construction | T |

Third release:  

| I | Construction | T |

Nth+ releases:  

| C | C | C | C |

**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

## Goals for the Inception Phase
- Form initial team
- Develop common project vision
- Align with enterprise direction
- Explore initial scope
- Identify initial technical strategy
- Develop initial release plan
- Form work environment
- Secure funding
- Identify risks

## Goals for Construction Phase Iterations
- Produce a potentially consumable solution
- Address changing stakeholder needs
- Move closer to deployable release
- Improve quality
- Prove architecture early

## Goals for the Transition Phase
- Ensure the solution is consumable
- Deploy the solution

### Ongoing Goals
- Fulfill the project mission
- Grow team members
- Address risk
- Improve team process and environment
- Leverage and enhance existing infrastructure

---

## Goal Driven Approach

![Diagram of Goal Driven Approach]

- **Goal**
- **Issue**
- **Default Option**
- **Advantages**
- **Disadvantages**
- **Considerations**

### Form the Initial Team
- Source
- Team size
- Team structure
- Team members
- Geographic distribution
- Supporting the team
- Availability

### Co-located
- Partially dispersed
- Fully dispersed
- Distributed subteams

---

© Scott Ambler + Associates
**Goal: Develop Common Vision**

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

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

- **Level of Agreement**
  - Consensus
    - General agreement
    - Dictated
    - None

- **Formality**
  - Statement of intent
    - Informal agreement
    - Contract

---

**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

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

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

A Disciplined Agile Manifesto
An Overview of Disciplined Agile Delivery (DAD)
DAD Roles
The Inception Phase
The Construction Phase
• The Transition Phase
• Governing Disciplined Agile Teams
• Comparing the DAD and SAFe Frameworks

Scrum Roles

😊 Scrum Master
😊 Product Owner
😊 Team Member
Disciplined Agile Delivery (DAD) Roles

Primary Roles
- Team Lead
- Product Owner
- Team Member
- Stakeholder
- Architecture Owner

Secondary Roles (for scaling)
- Independent Tester
- Specialist
- Domain Expert
- Technical Expert
- Integrator

Stakeholder
- Stakeholder is more than a customer
- Anyone impacted by the outcome of the system
- Types of stakeholders
  - End users: Users of the system
  - Principals: Decision makers that pay for and put the system to use
  - Partners: People who make the system work in production
  - Insiders: Members of the development team and people who provide business and technical services to the team
Product Owner

- The Stakeholder “proxy”
- Go-to person for information on the solution requirements
- Prioritizes all work for the team
- Participant in modeling and acceptance testing
- Has access to expert stakeholders
- Facilitates requirements envisioning and modeling
- Educates team in business domain
- May demonstrate solution to key stakeholders
- Monitors and communicates status to stakeholders
- Negotiates priorities, scope, funding, and schedule

Team Member

- Is a cross-functional, generalizing specialist
- On small teams every team member is typically a developer, but on larger teams non-developers may appear
- Volunteers to do any work that allows the team to most efficiently deliver the work committed to for the iteration
- Seeks to both learn about other specialties as well as coach others on their own specialty
- Goes to the product owner for domain information and decisions
- Works with the architecture owner to evolve the architecture
- Follows enterprise conventions and leverage and enhance the existing infrastructure
Team Lead

• Responsible for the effectiveness and continuous improvement of the team’s process
• Facilitates close collaboration between team members
• Keeps the team focused on the project vision and goals
• Removes impediments for the team and escalates organizational impediments
• Protects the team from interruptions and external interferences
• Maintains honest communication between everyone on the project
• Coaches others in the use of agile practices
• Prompts the team to discuss and think through issues when they are identified
• Facilitates decision making (but does not make decisions or mandate internal team activity)

Architecture Owner

• Guides the creation and evolution of the solution’s architecture
• Mentors and coaches team members in architecture practices and issues
• Understands the architectural direction and standards of your organization and ensures that the team adheres to them
• Ensures the system will be easy to support by encouraging appropriate design and refactoring
• Ensures that the system is integrated and tested frequently
• Has the final decision regarding technical decisions, but doesn’t dictate them
• Leads the initial architecture envisioning effort
Interacting with Other Teams

- Other agile delivery teams
- Other non-agile development teams
- Quality Assurance (QA) and testing
- Technical writers
- User Experience (UX) experts
- Enterprise architects
- Governance body
- Project Management Office (PMO)
- Data management/administration
- Reuse engineers

Agenda

- A Disciplined Agile Manifesto
- An Overview of Disciplined Agile Delivery (DAD)
- DAD Roles
- The Inception Phase
- The Construction Phase
- The Transition Phase
- Governing Disciplined Agile Teams
- Comparing the DAD and SAFe Frameworks
The Inception Phase

Inception Phase Goals

- **Inception Phase**
  - Form initial team
  - Develop common project vision
  - Align with enterprise direction
  - Explore initial scope
  - Identify initial technical strategy
  - Develop initial release plan
  - Form work environment
  - Secure funding
  - Identify risks

- **Ongoing**
  - Fulfill the project mission
  - Improve team process and environment
  - Grow team members
  - Leverage and enhance existing infrastructure
  - Address risk
The Inception Phase

 Coordinate

 Up to a few hours

 Collaborate

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

 Conclude

 Up to a few hours

 Stakeholder consensus

- Initiate team
- Schedule stakeholders for envisioning sessions
- Build team
- Requirements envisioning
- Architecture envisioning
- Consider feasibility
- Align with enterprise strategy
- Release planning (initial)
- Develop shared vision
- Setup environment
- Light-weight milestone review
- Communicate vision to stakeholders

Project Selected

Industry Data: Inception Activities

Initial Requirements Modeling
Initial Architecture Modeling
Justify Project
Initial Estimate
High-Level Release Schedule

Source: Ambysoft 2009 Agile Project Initiation Survey
Goal: Align With Enterprise Direction

- Align With Enterprise Direction
  - Adopt Common Guidelines
    - High level
      - Detailed
        - Optional
          - Enforced
            - None
  - Adopt Common Templates
    - Minimal
      - Comprehensive
        - None
  - Coordinate With Enterprise Teams
    - Collaborative
      - Continuous
        - Gated
          - Formal
            - None
  - Reuse Existing Infrastructure
    - Managed – Collaborative
      - Managed – Formal
        - Ad hoc
          - None
  - Adopt Governance Strategy
    - Agile
      - Traditional
        - None

Initial Requirements
Goal: Explore the Initial Scope

- Level of Detail
  - Goals driven
  - Requirements envisioning (light specification)
  - BRUF (Detailed specification)
  - None

- Usage modeling
  - Domain modeling
  - Process modeling
  - User interface modeling
  - Non-functional requirements

- View Types
  - Informal modeling sessions
  - Formal modeling sessions
  - Interviews
  - None

- Modeling Strategy
  - Work item pool
  - Work item stack
  - Scrum product backlog
  - Formal change management
  - None

- Work Item Management Strategy
  - Acceptance criteria
    - Explicit list
    - Technical stories
    - None

- Non-Functional Requirements

Choosing the Appropriate Level of Initial Detail

- BRUF (detailed specifications)
- Requirements envisioning (lightweight specifications)
- Goals driven
- No modeling at all
Work Item Management:

Work Item Stack – Ordered by Risk and Value

- Each iteration implement the highest-priority work items
- Each new work item is prioritized and added to the stack
- Work items may be reprioritized at any time
- Work items may be removed at any time

Goal: Identify Initial Technical Strategy

- Level of Detail
  - High-level overview
  - Detailed interface
  - Detailed end-to-end
  - None
- View Types
- Technology
  - Business Architecture
  - User Interface (UI)
- Informal modeling sessions
- Formal modeling sessions
- Single candidate architecture
- Multiple candidate architectures
- Extend existing solutions
  - Configure a COTS package
  - Extend a COTS package
  - Build from scratch
Identify Technical Unknowns/Risks

- As you identify your initial technical strategy you are likely to identify issues or complexities that you don’t know the answer for
- For example:
  - You believe you can access data from a legacy system, but aren’t sure of the details
  - Will a new technology platform will work as the vendor claims?
  - You have a requirement for a given level of throughput, e.g. 5,000 transactions a second
  - Your team has never worked with your organization’s web services framework before
- You will want to explore these risks early in Construction by:
  - Proving the architecture via working code
  - Architecture spikes to explore an aspect of your solution

Initial Release Planning

- The vast majority of agile teams invest some initial time in planning their release

- Considerations:
  - Schedule
  - Cost
  - Work allocations
  - How many releases and timing of each?
  - Length of iterations?

- 73% of agile teams produce an initial high-level estimate, and 18% of agile teams produce a detailed one*

- 77% of agile teams produce an initial high-level schedule, and 12% of agile teams produce a detailed one*

* Source: Ambysoft 2009 Agile Project Initiation Survey
Goal: Develop Initial Release Plan

Develop Initial Release Plan

Scheduling Strategy

Source

Scope

Self-organizing team
Manager facilitated
Manager driven

Portfolio
Product/solution
Release
Iteration
Daily

Predictive light
Adaptive light
Adaptive detailed
Predictive detailed
None

Production releases
Phase duration
Internal releases
Iteration length

Educated guess by experienced individual
Educated guess by team
Similar sized items
Planning poker
Formal point counting
Cost set by stakeholders
None

Depicting your Release Schedule with a Gantt Chart

<table>
<thead>
<tr>
<th>Milestone/dependency</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Start</td>
<td>Mar 5</td>
</tr>
<tr>
<td>Stakeholder consensus</td>
<td>Mar 23</td>
</tr>
<tr>
<td>Proven architecture</td>
<td>Apr 6</td>
</tr>
<tr>
<td>Project viability</td>
<td>Jun 1</td>
</tr>
<tr>
<td>Security framework delivered</td>
<td>Jun 15</td>
</tr>
<tr>
<td>Project viability</td>
<td>Jul 27</td>
</tr>
<tr>
<td>Project viability</td>
<td>Sep 21</td>
</tr>
<tr>
<td>Sufficient functionality</td>
<td>Oct 5</td>
</tr>
<tr>
<td>Production ready</td>
<td>Nov 2</td>
</tr>
</tbody>
</table>
Examples of Release Patterns

Classic DAD Pattern

One multi-year project with a single release at the end

One Inception phase, variable length Construction phases

Lean approach of continuous & frequent Transition to customers, periodic Inception phase

Goal: Identify Risks
Choosing Iteration Lengths

The average construction iteration length is 2.3 weeks*

1 week or less 15%
2 weeks 51%
3 weeks 15%
4 weeks 10%
> 4 weeks 2%

Heuristics:
- Shorter is generally better than longer
- Teams at scale may require slightly longer iterations

* Source: Amyssoft November 2010 Agile State of the Art Survey

Inception Goal: Form the Initial Team

- Source: Existing team from another product or new
- Team Size: Small, Medium, Large
- Team Structure: Single team, Component teams, Feature teams, Internal open source
- Whole team: Specialized team, Generalizing specialists, Specialists
- Co-located: Partially dispersed members, Fully dispersed members, Distributed subteams
- Supporting the Team: Coaching, Mentoring, Training, Stakeholder access
- Availability: Dedicated, Ongoing part-time, As needed/available
Goal: Form Work Environment

- Physical Environment
  - Work room
    - Open work area
    - Caves and commons
    - Near-located cubicles
    - Far-located members
  - Development tools
    - Governance tools
  - Commercial tools
    - Open source tools
    - Integrated tool suite
    - Point-specific tools
    - Automated tools
    - Manual tools
  - Choose toolset

- Virtual Environment

Organizing Physical Work Environments

- Dedicated space
- Significant whiteboard space
- Digital camera
- Agile modeling supplies
- A bookshelf or storage cabinet
- Large table
- Wall space to attach paper
- Projector
- Reference books
- Food, toys...
### Milestone: Bringing Stakeholders to Agreement around your Vision

- Inception is complete and you can enter the Construction phase when:
  - Your stakeholders agree that it makes sense to proceed based upon the achievable scope, schedule, budget, constraints, and other criteria related to your business case
  - The risks have been identified and seem tolerable
  - There is agreement on using a minimalist and agile process for building the solution
  - The team and environment have been set up or are in the process of being so, that supports collaborative teamwork
  - The process and governance strategies have been agreed to by both your team and your stakeholders
Goal: Secure Funding

- Funding Strategy
  - T&M plus performance bonus
  - Time and materials (T&M)
  - Stage gate
  - Fixed price/cost (ranged)
  - Fixed price/cost (exact)

- Secure Funding
- Access Strategy
- IT funding pool
  - Informal request
  - Formal request

Goal: Develop Common Vision

- Vision Strategy
  - Collaborative
    - Stakeholder driven
    - Sponsor driven
    - Team driven
  - Information radiators
    - Light-weight
    - Detailed
    - None

- Level of Detail
  - Consensus
    - General agreement
    - Dictated
    - None

- Level of Agreement
- Formality
  - Statement of intent
    - Informal agreement
    - Contract
What’s in a Project Vision?

- Also known as project charter or business case

- Typically outlines:
  - Goals of the project team and who is on it
  - High-level scope of the current release
  - Technical overview of the solution
  - Outline of the plan to do the required work
  - Could include feasibility information

- Could also describe:
  - Description of business problem being addressed
  - High-level schedule and estimates
  - Key milestones
  - Stakeholders
  - Funding models
  - Project risks and constraints
  - Process/method used (eg DAD), governance strategy
  - Key assumptions

Inception Patterns and Anti-Patterns

- Patterns that can increase your likelihood of success
  - Have a short but sufficient Inception phase
  - Ranged estimates
  - Minimal but sufficient documentation

- Anti-patterns
  - No support for skills development
  - No support for dedicated facilities
  - Autocratic project management practices
  - Jumping into Construction
  - Overly detailed work products
  - Analysis paralysis
Agenda

- A Disciplined Agile Manifesto
- An Overview of Disciplined Agile Delivery (DAD)
- DAD Roles
- The Inception Phase
- The Construction Phase
- The Transition Phase
- Governing Disciplined Agile Teams
- Comparing the DAD and SAFe Frameworks

Construction Phase Goals

- Construction Phase
  - Produce a potentially consumable solution
  - Address changing stakeholder needs
  - Move closer to deployable release
  - Improve quality
  - Prove architecture early

- Ongoing
  - Fulfill the project mission
  - Improve team process and environment
  - Grow team members
  - Leverage and enhance existing infrastructure
  - Address risk
The Construction Phase

- Initial Architectural Vision
- Identify requirements and release plan
- Initial planning session to select work items and identify tasks for current Iteration
- Iteration planning session to select work items and identify work tasks for current Iteration
- Potentially Consumable solution
- Iteration Review & retrospective: Demo to stakeholders, determine strategy for next Iteration, and learn from your experiences
- Funding
- Feedback
- Enhancement Requests and Defect Reports
- Construction
  - Many short Iterations producing a potentially consumable solution each Iteration
  - Proven Architecture
  - Project viability (several)
  - Sufficient functionality

© Scott Ambler + Associates

The Construction Phase

- Prove the architecture works via end-to-end working slice of the solution
- Coordinate
  - Typical: 1 Iteration
  - Worst case: Many iterations
- Stakeholder Consensus
- Several Iterations
- Several iterations
- Proven Architecture
- Conclude
  - Ideally: Several hours
  - Sufficient Functionality
- 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
  - Determine sufficiency
  - Harden the solution

© Scott Ambler + Associates
A Construction Iteration

- Iteration planning
- Iteration modeling

“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-ado development
- Look-ahead planning

- Iteration demo
- Retrospective
- Release planning (sprints)
- Determine “go forward” strategy

Coordinate

- 2 hours for each week of the iteration length

Collaborate

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

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
- Conclude

Ideally: Not a concern

Start of day

Up to 15 minutes

Typical: 5-6 hours

End of day

A Typical Day of Construction
Initiating a Construction Iteration

- Agile planning is different from traditional planning
  - Detailed planning occurs throughout the project, typically an iteration at a time
  - The team self-organizes and plans its own work
  - Team coordination occurs daily
  - The high-level release plan is updated appropriately throughout the project

- The iteration starts with a short planning workshop
  - Iteration planning
  - Iteration modeling

Visualizing Your Plan
Goal: Prove Architecture Early

End-to-end working skeleton
Architectural spikes
Solution bake-off
Pilot testing
Stakeholder demos
Informal reviews
Formal reviews

Informal reviews
Formal reviews

The Risk-Value Lifecycle

© Scott Ambler + Associates
### Critical Agile Practices

- Continuous documentation
- Non-solo work
- Iteration burndown charts
- Change management
- Configuration management
- Architectural spikes
- Test-driven development (TDD)
- Continuous integration (CI)
- Continuous deployment (CD)
- Look-ahead planning
- Look-ahead modeling
- Parallel independent testing
- Reviews
Estimating: Ranged Release Burndowns

- A ranged estimate of number of iteration required to complete work
- Range of uncertainty decreases as project progresses

Demonstrate the Solution to Key Stakeholders

- Iteration demonstration
- On-demand demonstrations
- All hands demonstrations
- None
Learn from Your Experiences

- Retrospectives
- Structured surveys
- Measured improvement
- Project post mortem
- Ad hoc process improvement
- None

Deploy your Current Build

- Deployment to appropriate environments:
  - Demonstration
  - Pre-production testing
  - Production
Determine Strategy Moving Forward

- Sources
  - Input from your iteration demo
  - The team
  - Usage statistics

- How to proceed?*
  - Persevere with the current strategy
  - Run a split test
  - Pivot in a new direction
  - Cancel the project

- Deciding how to proceed
  - Team meeting
  - Lightweight milestone review
  - Formal milestone review

* From ‘The Lean Startup’, Eric Ries

Construction Patterns

- The team can be reliably depended on to demonstrate increments of software at the end of each iteration

- Team members finish their tasks ahead of schedule and ask to help others with their tasks

- Iteration dates never move

- Any stakeholder can expect to see a demonstration of working software at any time
Construction Anti-Patterns

- A work item list that is too big to easily manage and comprehend
- Inattention to risk mitigation
- Assuming that the architecture will work without proving it with code
- Assuming that an iterative approach alone ensures that you will build the right solution in an effective manner
- One or more of your team members are working on other projects in addition to yours
- A work item isn’t done
- Last iteration we planned for X points but delivered Y (less than X), but we still commit to X this iteration
- During the iteration we missed some tasks when iteration planning
- During the iteration we realized we missed a requirement that another depends on
- During iteration planning sessions, the product owner is reprioritizing the backlog or is uncertain of requirement details
- Defect counts are increasing in each iteration

Agenda

- A Disciplined Agile Manifesto
- An Overview of Disciplined Agile Delivery (DAD)
- DAD Roles
- The Inception Phase
- The Construction Phase
- The Transition Phase
- Governing Disciplined Agile Teams
- Comparing the DAD and SAFe Frameworks
The Transition Phase

- **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

- **Coordinate**
  - Ideally: Nothing
  - Typical: One hour per week of collaborate time
  - Sufficient Functionality

- **Collaborate**
  - Ideally: Nothing
  - Average: 4 weeks
  - Worst case: Several months

- **Conclude**
  - Ideally: Less than an hour
  - Worst case: Several months
  - Production Ready

- **Actual usage**
  - Delighted Stakeholders

---

The Transition Phase

- **Potentially Consumable Solution**
- **Release solution into production**

- **Enhancement Requests and Defect Reports**

- **Operate and support solution in production**

- **Transition**
  - One or more short iterations
  - Production ready
  - Delighted stakeholders
Transition Phase Goals

- Construction Phase
  - Ensure the solution is consumable
  - Deploy the solution

- Ongoing
  - Fulfill the project mission
  - Improve team process and environment
  - Grow team members
  - Leverage and enhance existing infrastructure
  - Address risk

Planning for the Transition Phase

- Some pre-planning activities during Construction ensure that the team is prepared going into Transition
  - Is paperwork required prior to deployments?
    - E.g. Request For Change (RFCs)
  - Do elements of production infrastructure need to be procured and setup?
  - Are there final user acceptance testing tasks to be done?
  - Do disaster recovery plans need to be updated?
  - Do changes need to be coordinated with other projects?
  - Is there support documentation required?
  - Are sign-offs required?
  - Does your team need to be on call?
Goal: Ensure Solution is Consumable

- Production Readiness
  - Finalize transition plan
  - End of lifecycle testing
  - End of lifecycle fixing
  - Deployment testing
  - Data migration preparation
  - Pilot/beta testing
  - Finalize documentation

- Stakeholder Readiness
  - Communicate deployment
  - Train/educate stakeholders
  - Prepare support environment

Goal: Deploy the Solution

- Release
  - Close out existing transactions
  - Backup existing solution
  - Restore existing solution
  - Migrate source data
  - Deploy solution components
  - Deployment testing
  - Make solution available
  - Enable support system
  - Communicate deployment

- Validate Release
  - Production deployment testing
  - Stakeholder satisfaction survey
  - None
Transition Phase Anti-Patterns

- Thinking Transition is a “hardening” phase
- Not having a production like environment for integration, acceptance, and deployment testing
- Not having an agreed upon set of entry criteria for going “live”
- Requests for new functionality
- Release the system to unprepared users
- Lengthy integration and user acceptance testing (UAT) cycle in Transition
- Transferring responsibility for maintaining the system to a maintenance group
- Moving all of your developers to another project at the end of Construction
- Not investing in stakeholder training
- Believing that installation is going to be easy

Agenda

- A Disciplined Agile Manifesto
- An Overview of Disciplined Agile Delivery (DAD)
- DAD Roles
- The Inception Phase
- The Construction Phase
- The Transition Phase
- Governing Disciplined Agile Teams
- Comparing the DAD and SAFe Frameworks
Governance Should Address a Range of Issues

- Team roles and responsibilities
- Individual roles and responsibilities
- Decision rights and decision making process
- Governing body
- Exceptions and escalation processes
- Knowledge sharing processes
- Metrics strategy
- Risk mitigation
- Reward structure
- Status reporting
- Audit processes
- Policies, standards, and guidelines
- Artifacts and their lifecycles

Aspects of Effective Agile Governance

- Trust and respect are the foundation of effective governance
- Be stakeholder driven
- Collaboratively define your governance strategy
- Be transparent
- Motivate, don’t dictate
- Enable, don’t enforce
- Optimize the “IT whole”, not the “governance part”
- Optimize corporate performance
- Collaboratively set reasonable guidance
- Collaboratively define rights and responsibilities
- Be suitable to task
- Automate wherever possible
DAD Milestones

<table>
<thead>
<tr>
<th>Milestone</th>
<th>Fundamental Question Asked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stakeholder consensus</td>
<td>Do stakeholders agree with your strategy?</td>
</tr>
<tr>
<td>Proven architecture</td>
<td>Can you actually build this?</td>
</tr>
<tr>
<td>Project viability</td>
<td>Does the project still make sense?</td>
</tr>
<tr>
<td>Sufficient functionality</td>
<td>Does it make sense to release the current solution?</td>
</tr>
<tr>
<td>Production ready</td>
<td>Will the solution work in production?</td>
</tr>
<tr>
<td>Delighted stakeholders</td>
<td>Are stakeholders happy with the deployed solution?</td>
</tr>
</tbody>
</table>

DAD Practices that Support Governance

- “Standard” agile practices:
  - Coordination meeting
  - Iteration demonstrations
  - All-hands demonstrations
  - Retrospective
  - Information radiators/Visual management

- Disciplined practices:
  - Risk-value lifecycle
  - Explicit light-weight milestones
  - Follow enterprise development guidelines
  - Work closely with enterprise professionals
  - Development intelligence via automated dashboards
Measuring Agile Teams

- Talk to people; don’t manage to the metrics
- Measure teams, not individuals
- Collect several metrics
- Trends are better than scalar values
- Empirical observation is important but limited
- Prefer automated metrics
- Some metrics must be gathered manually
- Prefer pull versus push reporting
- Beware scientific facades
- Measure the value of your metrics program
- Be prepared to educate people
- The value of many metrics diminishes over time
- If you collect no metrics at all you’re flying blind
- If you collect too many metrics you may be flying blinded

Potential Metrics

- Acceleration
- Activity time
- Age of work items
- Blocking work items
- Build health
- Business value delivered
- Change cycle time
- Code quality
- Defect density
- Defect trend
- Effort/cost projection
- Iteration burndown
- Lifecycle traceability
- Net present value (NPV)
- Ranged release burndown
- Release burndown
- Return on investment (ROI)
- Risk mitigation
- Stakeholder satisfaction
- Team morale
- Test coverage
- Time invested
- Velocity
Disciplined Agile Delivery Tutorial

Agenda

• A Disciplined Agile Manifesto
• An Overview of Disciplined Agile Delivery (DAD)
• DAD Roles
• The Inception Phase
• The Construction Phase
• The Transition Phase
• Governing Disciplined Agile Teams
• Comparing the DAD and SAFe Frameworks
Team success through “Disciplined” agile delivery at the project level
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

DAD Lifecycle: Basic/Agile

![DAD Lifecycle Diagram]

<table>
<thead>
<tr>
<th>Exception</th>
<th>Construction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stakeholder consensus</td>
<td>Project viability (satisfied)</td>
</tr>
<tr>
<td>Proven architecture</td>
<td>Sufficient functionality</td>
</tr>
<tr>
<td>Many short iterations producing a potentially consumable solution each iteration</td>
<td>Iteration planning session to select work items and identify work tasks for current iteration</td>
</tr>
<tr>
<td>Senior leadership and support, stakeholders invested</td>
<td>Iteration review &amp; retrospective, re-evaluate strategic direction, and make next iteration cycle better</td>
</tr>
<tr>
<td>Release to production, feedback from customers and users</td>
<td>Feedback from customers and users, enhancement requests and detect defects</td>
</tr>
</tbody>
</table>

![DAD Lifecycle Diagram]
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></td>
</tr>
<tr>
<td>- Develop common project vision</td>
<td>- Address changing stakeholder needs</td>
<td></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

Scaling/Tailoring Complexity Factors

- Geographic Distribution
  - Co-located
  - Global

- Organizational Distribution
  - Single
  - Outsourcing

- Project Type
  - Multiple Release
  - Single Release

- Team Size
  - 2
  - 1000s

- Team Culture
  - Agile
  - Rigid

- Organizational Culture
  - Agile
  - Rigid

- Domain Complexity
  - Straightforward
  - Very complex

- Technical Complexity
  - Straightforward
  - Very complex

- Compliance
  - None
  - Life critical

© Scott Ambler + Associates
Disciplined Agile Delivery: The Foundation for Scaling Agile

<table>
<thead>
<tr>
<th>Organizational Distribution</th>
<th>Team Culture</th>
<th>Organizational Culture</th>
<th>Domain Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geographic Distribution</td>
<td>Team Size</td>
<td>Compliance</td>
<td>Technical Complexity</td>
</tr>
<tr>
<td>Outside In Dev.</td>
<td>“Traditional”</td>
<td>And more...</td>
<td></td>
</tr>
<tr>
<td>Unified Process</td>
<td>Agile Modeling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scrum</td>
<td>Kanban</td>
<td>Lean</td>
<td></td>
</tr>
</tbody>
</table>

Disciplined Agile Delivery (DAD)

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

Discussion: What Still Puzzles You?

- This tutorial has potentially presented you with a lot of new ideas
- It can be incredibly difficult to take in all of this material at once
- So... is there anything that still puzzles you about Disciplined Agile Delivery (DAD)?
Disciplined Agile Certification

www.DisciplinedAgileConsortium.org

Disciplined Agile Yellow Belt
– Indication that the person is new to disciplined agile but eager to learn
– Beginner certification

Disciplined Agile Green Belt
– Indication that the person is striving to be a professional
– Potential to be a junior coach
– Intermediate certification

Disciplined Agile Black Belt
– Indication that the person is an expert
– Often a senior coach, instructor, or agile transformation lead
– Expert certification

Give Feedback on the Sessions

1 Sign In: www.eclipsecon.org

2 Select Session Evaluate

3 Vote +1 0 -1
Thank You!

mark@sco7ambler.com
@mark_lines

DisciplinedAgileDelivery.com
DisciplinedAgileConsortium.org
ScottWAmbler.com

Recommended Resources