Agile Domains, Tools, and Techniques

Contents

Notes from Mike Griffith, *PMI-ACP Exam Prep,* Second Edition.

Skip to main content

Introduction

Domains

Domain	Weight	Sub-Domain
Agile Principles and Mindset	16%	
Value-Driven Delivery	20%	Define Positive Value Avoid Potential Downsides Prioritization Incremental Development
Stakeholder Engagement	17%	Understand Stakeholder Needs Ensure Stakeholder Involvement Manage Stakeholder Expectations
Team Performance	16%	Team Formation Team Empowerment Team Collaboration and Commitment
Adaptive Planning	12%	Levels of Planning Adaptation Agile Sizing and Estimation
Problem Detection and Resolution	10%	

Skip to main content

Tools and Techniques

Toolkit	Tool/Technique
Agile Analysis and Design	Product Roadmap
	User Stories/Backlog
	Story Maps
	Progressive Elaboration
	Wireframes
	Chartering
	Personas
	Agile Modeling
	Workshops
	Learning Cycle
	Collaboration Games
Agile Estimation	Relative sizing/story points/T-shirt sizing
	Wide ban Delphi/Planning Poker
	Affinity Estimation

Skip to main content

Toolkit	Tool/Technique
	Ideal Time
Communications	Information Radiator
	Team Space Agile Tooling
	Osmotic Communications for Co-located and/or Distributed Teams
	Two-way Communications (Trustwhorty, Conversation Driven)
	Socia-media based Communication
	Active Listening
	Brainstorming
	Feedback Methods
Interpersonal Skills	Emotional Intelligence
	Collaboration
	Adaptive Leadership
	Servant Leadership
	Negotiation
	Conflict Resolution

Skip to main content

Toolkit	Tool/Technique
Metrics	Velocity/Throughput/Productivity
	Cycle Time
	Lead Time
	EVM for Agile Projects
	Defect Rate
	Approved Iterations
	Work in Progress
Planning, Monitoring, and Adapting	Reviews
	Kanban Board
	Task Board
	Timeboxing
	Iteration and Release Planning
	Variance and Trend Analysis
	WIP Limits
	Daily Stand Ups

Skip to main content

Toolkit	Tool/Technique
	Burn down/up Charts
	Cumulative Flow Diagram
	Backlog Grooming/Refinement
	Product-feedback Loop
Process Improvement	Kaizen
	Five WHYs
	Retrospectives, Intraspectives
	Process Tailoring/Hybrid Models
	Value Stream Mapping
	Control Limits
	Pre-mortem (Rule Setting, Failure Analysis)
	Fishbone Diagram Analysis
Product Quality	Frequent Verification and Validation
	Definition of Done
	Continuous Integration

Skip to main content

Toolkit	Tool/Technique
	Testing, Including Exporatory and Usability
Risk Management	Risk Adjusted Backlog
	Risk Burn Down Graphs
	Risk-based Spike
	Architectural Spike
Value-Based Prioritization	ROI/NPV/IRR
	Compliance
	Customer Value Prioritization
	Requirements Reviews
	Minimal Viable Product (MVP)
	Minimal Marketable Feature (MMF)
	Relative Prioritization/Ranking
	MoSCoW
	Kano Analysis

Skip to main content

Agile Principles and Mindset

The Agile Mindset

Declaration of Interdependence (DOI)

- Written in 2005 by Agile Project Leadership Network
- Six Precepts:

> 1. We increase **return on investment** by making continuous flow of value our focus.
> 2. We **deliver reliable results** by engaging customers in frequent interactions and shared ownership.
> 3. We **expect uncertainty** and manage for it through iterations, anticipation, and adaptation.
> 4. We **unleash creativity** and innovation by recognizing that individuals are the ultimate source of value, and creating an environment where they can make a difference.
> 5. We **boost performance** through group accountability for results and shared responsibility for team effectiveness.
> 6. We **improve effectiveness and reliability** through situationally specific strategies, processes, and practices.

Skip to main content

The Agile Triangle

Methodology	Constraint	Variable
Predictive	Scope	Time, Cost
Agile	Time, Cost	Scope

The Agile Manifesto

Four Values

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

Twelve Principles

> Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Skip to main content

> Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.
>
> Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
>
> Business people and developers must work together daily throughout the project.
>
> Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
>
> The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
>
> Working software is the primary measure of progress.
>
> Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
>
> Continuous attention to technical excellence and good design enhances agility.
>
> Simplicity—the art of maximizing the amount of work not done—is essential.
>
> The best architectures, requirements, and designs emerge from self-organizing teams.
>
> At regular intervals, the team reflects on how to become more effective, then tunes and adjust sits behavior accordingly.

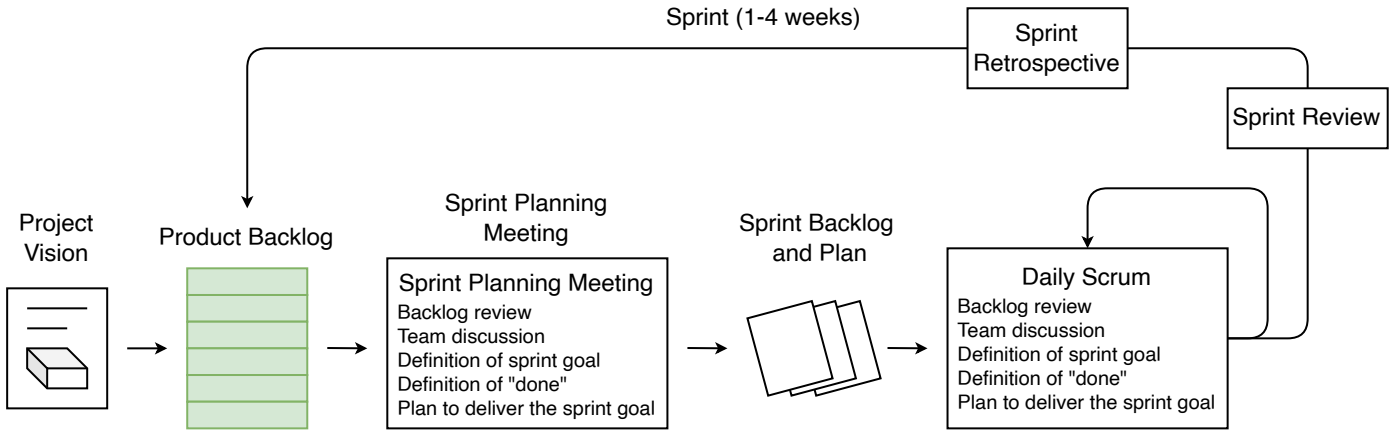
Skip to main content

Agile Methodologies

- Scrum
- Extreme Programming (XP)
- Lean Product Development
- Kanban
- Feature-driven Development (FDD)
- Dynamic Systems Development Method (DSDM)
- Crystal

Scrum

Process



Skip to main content

Principles

- Transparency
- Inspection
- Adaptation

Values

- Focus
- Courage
- Openness
- Commitment
- Respect

Sprints

- Sprint = timeboxed iteration of < 1 month
- No changes affecting the sprint goal are made throughout the sprint
- Scope can be clarified/renegotiated as new information becomes available
- Can be cancelled by Product Owner before timebox is over due to
    - goal becomes obsolete
    - change in business direction/technology conditions
    - sequence of Activities
        1. Sprint Planning Meeting

Skip to main content

    - Daily scrums
    - Sprint review meeting
    - Sprint retrospective meeting

Team Roles

- Product Owner
- Scrum Master
- Development Team

Activities (Events/Ceremonies)

- Product Backlog Refinement
- Sprint Planning Meeting
- Daily Scrums
- Sprint Reviews
- Sprint Retrospectives

Artifacts

- Product Increment
- Product Backlog
- Sprint Backlog

Skip to main content

Extreme Programming (XP)

Core Values

- Simplicity
- Communication
- Feedback
- Courage
- Respect

Team Roles

- Coach
- Customer
- Programmers
- Testers

Practices

- Whole Team
- Planning Games
- Small Releases
- Customer Tests

Skip to main content

- Code Standards
- Sustainable Pace
- Metaphor
- Continuous Integration
- Test-Driven Development
- Refactoring
- Simple Design
- Pair Programming

Lean Product Development

Core Concepts

- Eliminate waste
- Empower team
- Deliver fast
- Optimize the whole
- Build quality in
- Defer decisions
- Amplify learning

Seven Wastes

Skip to main content

- Extra processes
- Extra features
- Task switching
- Waiting
- Motion
- Defects

Kanban

To Do	In Progress	Done
G	C	A
H	D	B
I	E	
J	F	

Principles

- Visualize workflow
- Limit WIP
- Manage flow
- Make process policies explicit

Skip to main content

WIP Limits

$\downarrow$ WIP $\rightarrow \uparrow$ Team's productivity

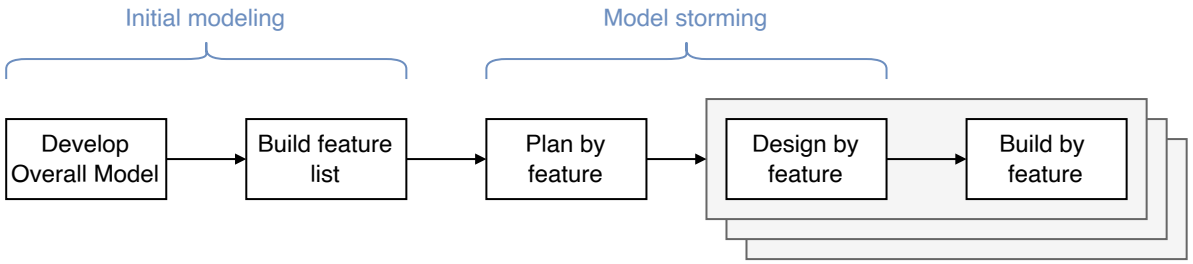
Little's Law: $Queue.Duration = m(Queue.Size)$

Backlog	Selected (4)	Develop (3)	Acceptance (2)	Deploy
L	I	F	E	A
M	J	G		B
N	K	H		C
O				D
P				
Q				

Feature-driven Development (FDD)

Process

Skip to main content



FDD Process

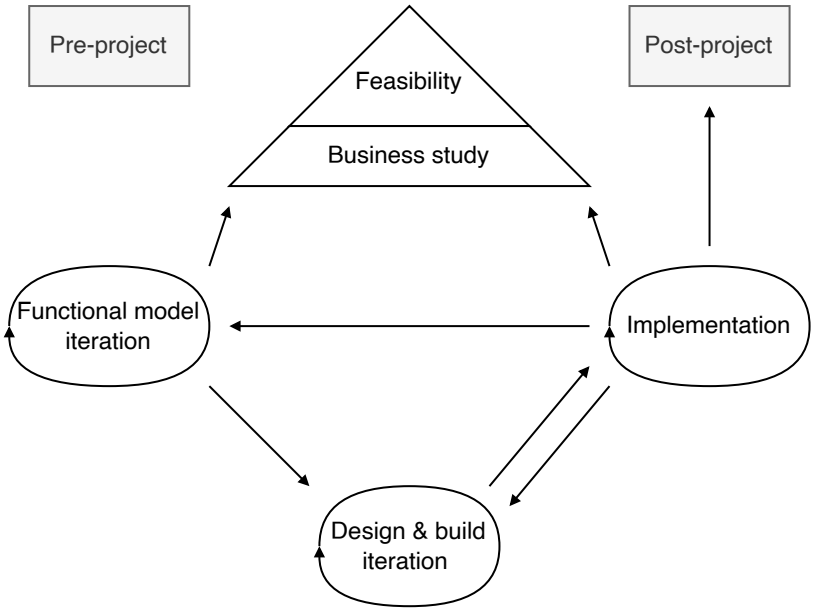
Practices

- Domain object modeling
- Developing by feature
- Individual class (code) ownership
- Feature teams
- Inspections
- Configuration management
- Regular builds
- Visibility of progress/results

Dynamic Systems Development Method (DSDM)

Process

Skip to main content



DSDM Process

Principles

- Focus on the business needs
- Deliver on time
- Collaborate
- Never compromise quality
- Build incrementally from firm foundations
- Develop iteratively
- Communicate continuously and clearly

Skip to main content

Crystal

Crystal = family of situationally specific, customzied methodologies coded by color names

$$Criticality = f(Defect.Impact)$$

Criticality	**Clear**	**Yellow**	**Orange**	**Red**	**Magenta**
Life	L6	L20	L40	L100	**L200**
Essential funds	E6	**E20**	**E40**	**E100**	**E200**
Discretionary funds	**D6**	**D20**	**D40**	**D100**	**D200**
Comfort	**C6**	**C20**	**C40**	**C100**	**C200**
Team size	1-6	7-20	21-40	41-100	101-200

Agile Leadership

Align project objectives with personal objectives to improve productivity

Skip to main content

Management versus Leadership

Management Focus	Leadership
Tasks/things	People
Control	Empowerment
Efficiency	Effectiveness
Doing things right	Doing the right things
Speed	Direction
Practices	Principles
Command	Communication

Servant Leadership

Duties

- Shield the team from interruptions
- Remove impediments to progress
- Communicate
- “Carry food and water”

Skip to main content

Value-Driven Delivery

Assessing Value

Financial Assessment Metrics

Return on Investment (ROI)

Formula: $ROI = Investment.Benefits / Investment.Cost$

Interpretation: $ROI > 1$

Present Value (PV)

Formula: $PV = FV_t / (1 + r)^t$

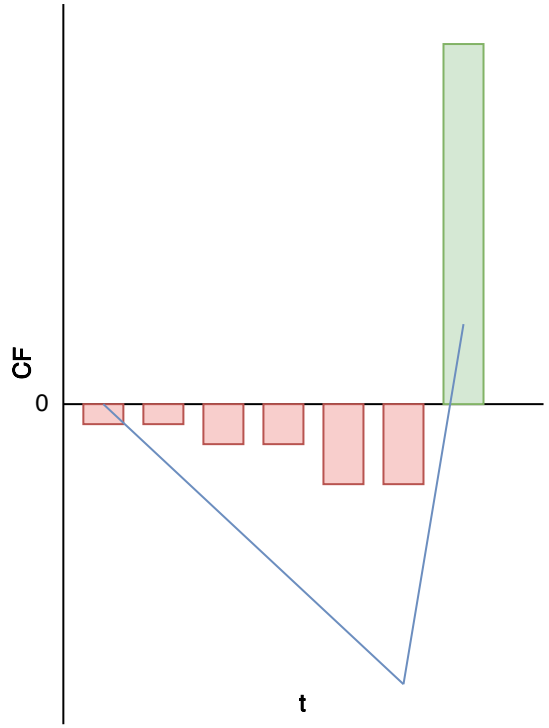
Net Present Value (NPV)

Formula: $NPV = \sum_{t=0}^{T} CF_t / (1 + r)^t$

Interpretation:

- IF $NPV > 0$ THEN accept ELSE reject
- Select project with **higher** NPV

Skip to main content



NPV calculation

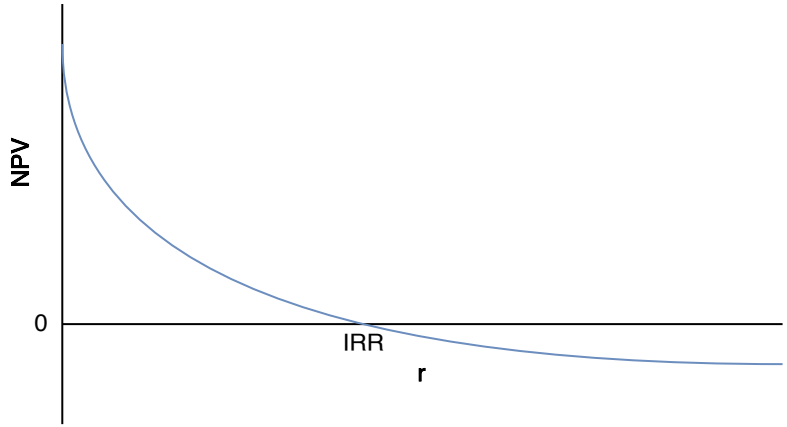
Internal Rate of Return (IRR)

Formula: $IRR = r : NPV(r) = 0$

Interpretation:

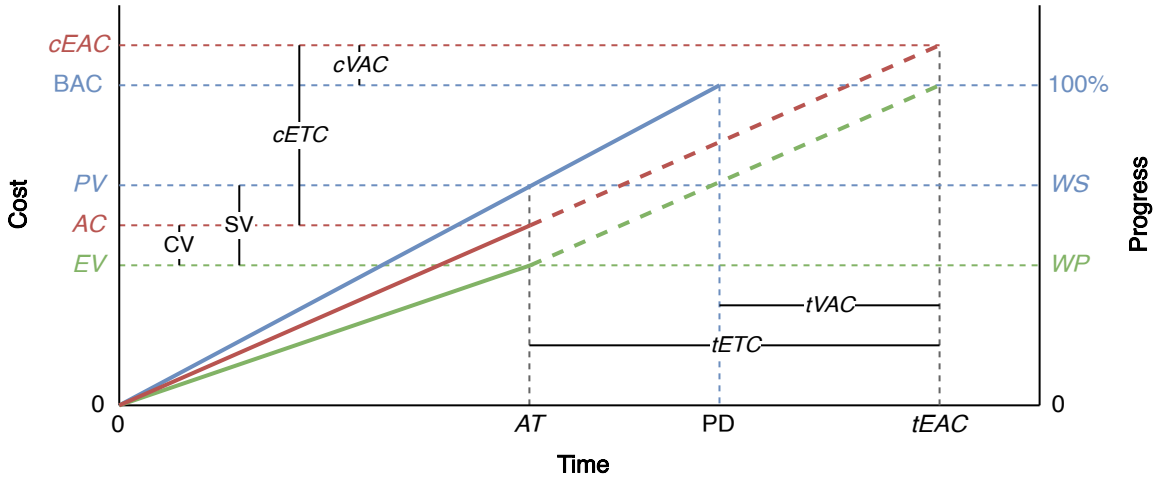
- IF $IRR > r$ THEN accept ELSE reject
- Select project with **higher** IRR

Skip to main content



IRR graphical derivation

Earned Value Management



Skip to main content

Symbol	Formula	Name
AT		Actual Time
WS		Work Scheduled
WP		Work Performed
BAC		Budget at Completion
PD		Planned Duration
AC		Actual Cost
PV	$\mathrm{BAC} \cdot WS$	Planned Value
EV	$\mathrm{BAC} \cdot WP$	Earned Value
CV	$EV - AC$	Cost Variance
SV	$EV - PV$	Schedule Variance
CPI	EV / AC	Cost Performance Index
SPI	EV / PV	Schedule Performance Index
$cEAC$	BAC $\mathrm{BAC} - CV$ BAC / CPI	Cost Estimate at Completion

Skip to main content

Symbol	Formula	Name
$tEAC$	PD PD/SPI	Time Estimate at Completion
$cETC$	$cEAC - AC$	Cost Estimate to Complete
$tETC$	$tEAC - AT$	Time Estimate to Complete
$cVAC$	$cEAC - \mathrm{BAC}$	Cost Variance at Completion
$tVAC$	$tEAC - \mathrm{PD}$	Time Variance at Completion

Agile Project Accounting

- Break down product/service into MVP
- Deliver MVP asap
- Exploit opportunities for early benefits by using part of the product/service while completing the remainder

Key Performance Indexes

- Rate of Progress
- Remaining Work
- Likely Completion Date
- Likely Costs Remaining

Skip to main content

Regulatory Compliance

- Regulations $\rightarrow$ safety
- A project that is subejct to regulatory compliance require special documentation to prove that required practices were followed
- Approaches for integrating regulatory compliance:
  - Doing compliance work **during** product development to keep them linked and relevant
  - Doing compliance work **after** product development to avoid rework

Prioritizing Value

Customer-Valued Prioritization

Work on items that maximize value delivered to customer first

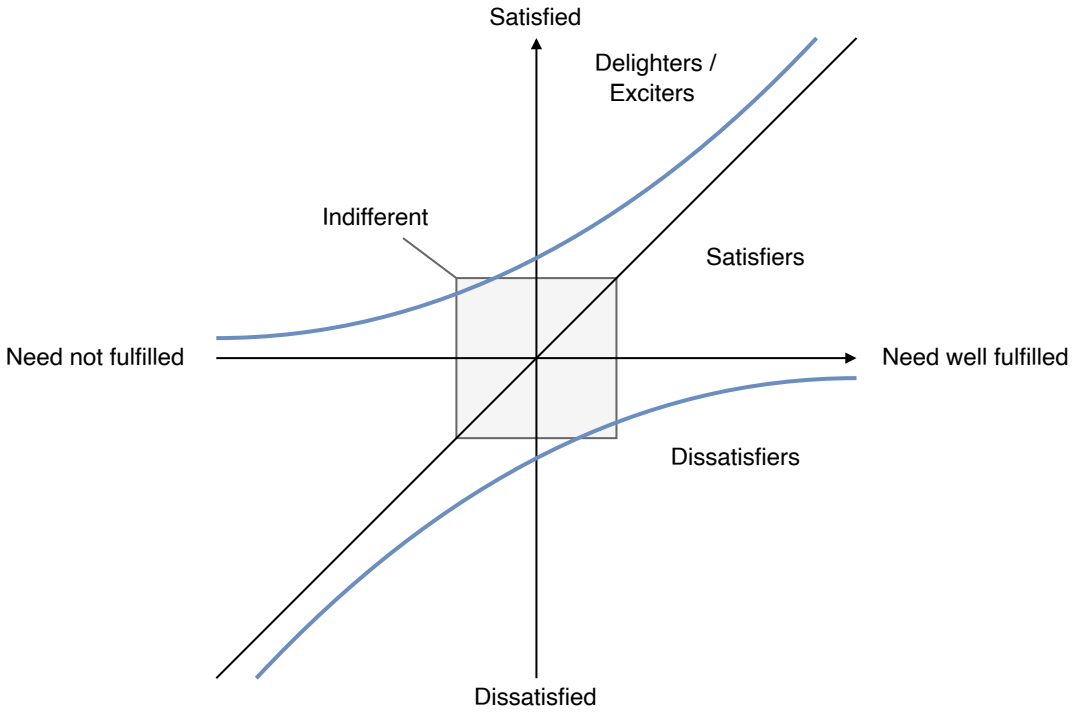
Prioritization Schemes

MoSCoW

Definition	Priority
Must have	Top
Should have	Medium
Could have	Low
Won’t have this time	Null

Kano Analysis

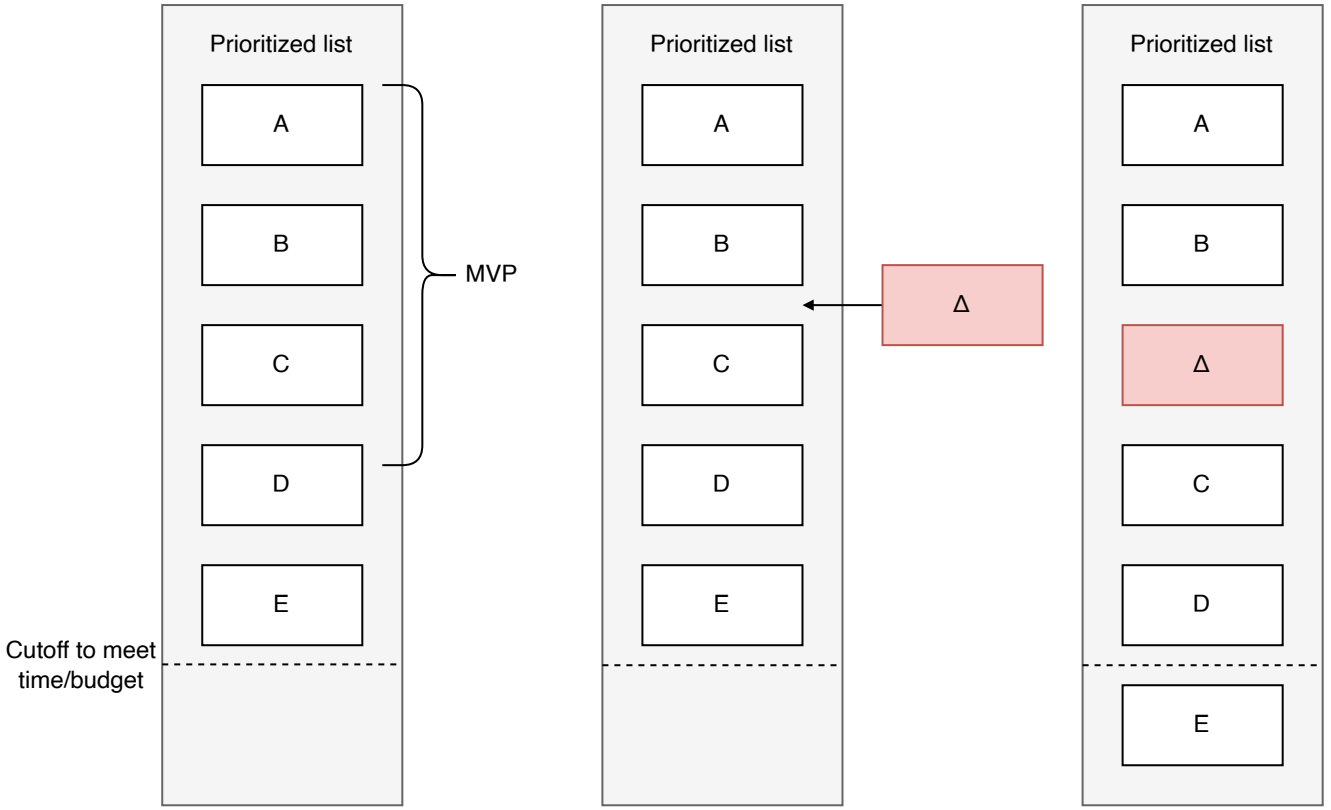
Skip to main content



Kano Analysis

Skip to main content

Relative Prioritization/Ranking

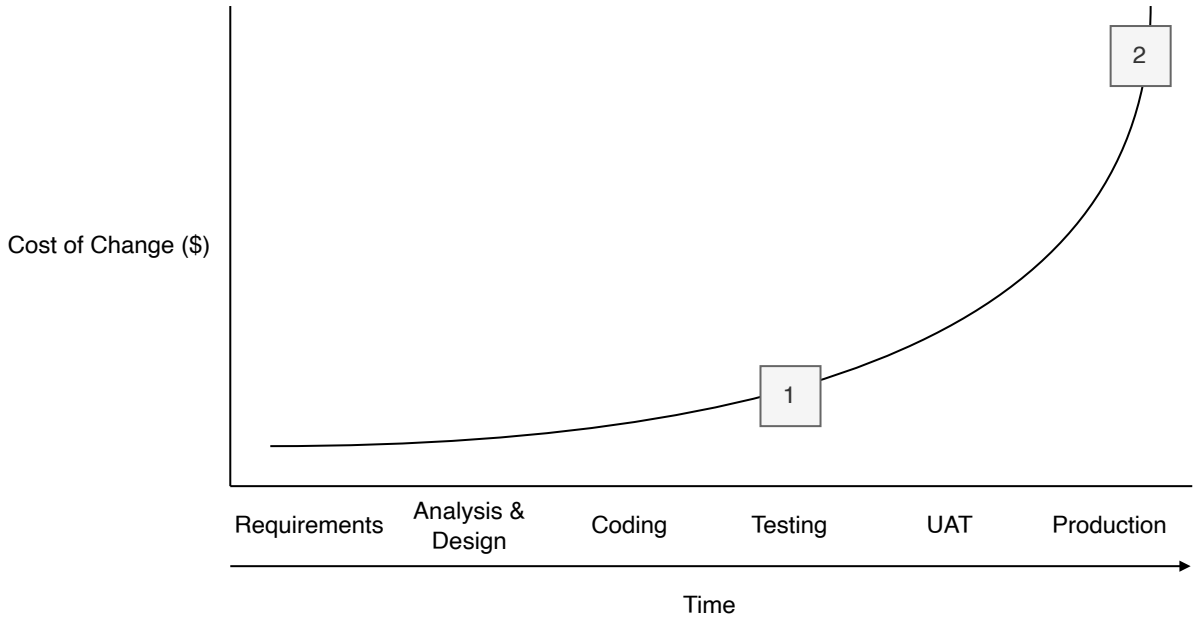


Incorporating changes into a relative priority list

Deliver Incrementally

Delivering the “plain-vanilla” version of a product/service allows realizing benefits to get an early ROI

Skip to main content



Cost of Change (Image copyright Scott W. Ambler, www.agilemodeling.com)

Minimum Viable Product (MVP)

MVP = Package of functionality that is complete enough to be useful to the users or the market, yet still small enough that it does not represent the entire project

Agile Tooling

Prefer low tech, high touch tools over sophisticated computerized models

Skip to main content

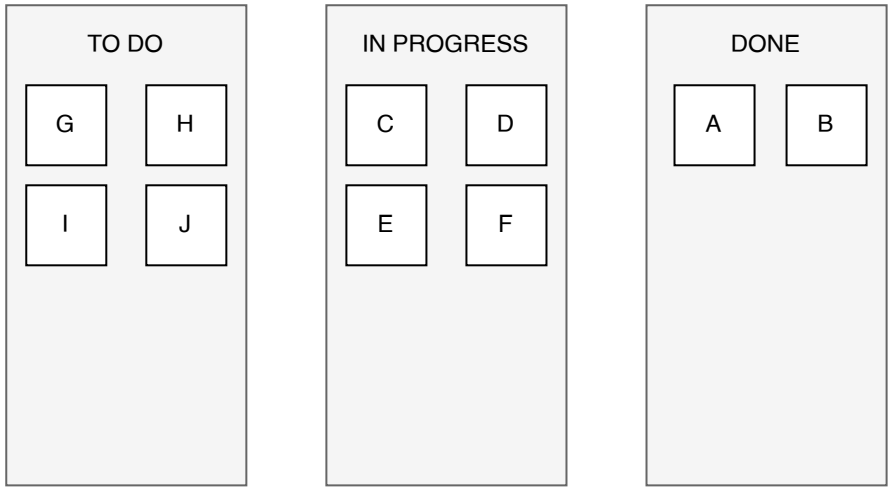
Task/Kanban Boards

Work in Progress (WIP)

- Work started but not started yet
- Excessive WIP:
  - consumes investment capital and delivers no ROI until converted into product/service
  - hides bottlenecks/inefficiencies
  - increases probability of rework

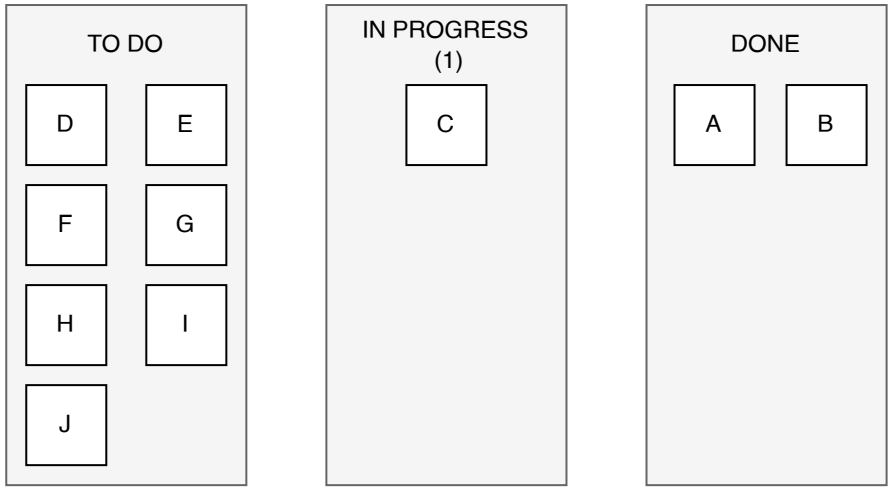
WIP Limits

Set limit to WIP (to Task/Kanban Board) $\rightarrow$ Optimize ~~resource utilization~~ throughput

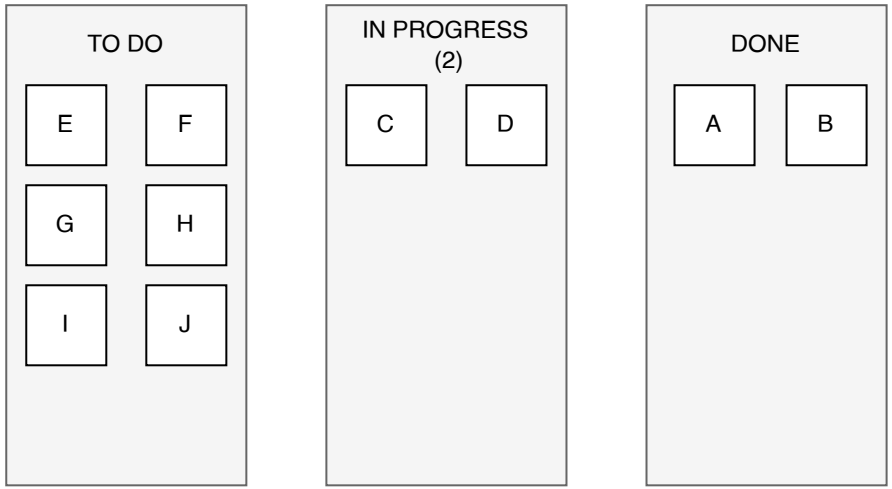


Kanban board without limits

Skip to main content



Kanban board with limit too strict

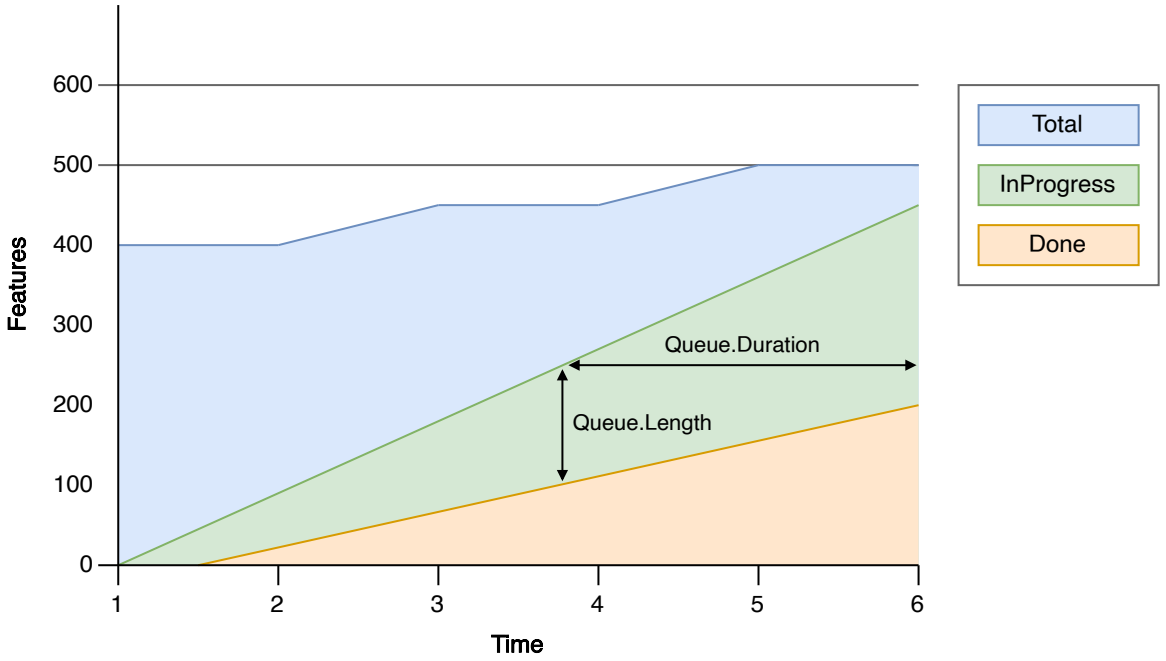


Kanban board with right limit

Skip to main content

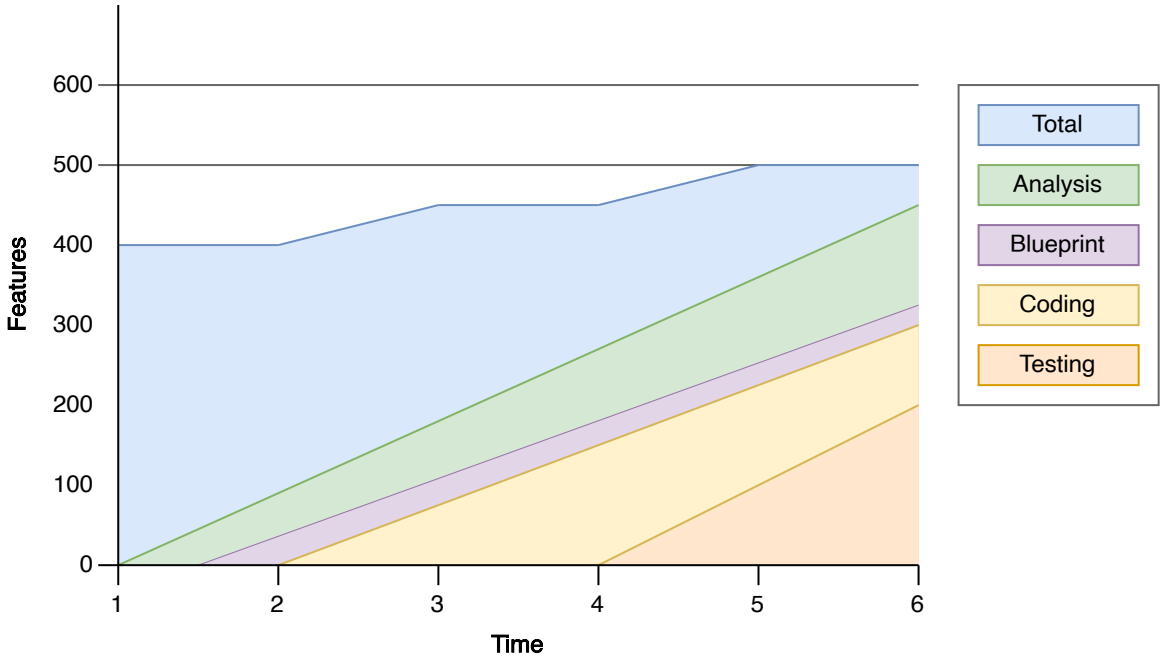
Cumulative Flow Diagram (CFDs)

Used for tracking and forecasting delivery of value



CFD

Skip to main content



Detailed CFD

Little's Law

$Queue = InProgress - Done$

Bottleneck and Theory of Constraints (TOC)

Figure

Skip to main content

(To Do) Agile Contracting

Verifying and Validating Value

(To Do) Frequent Verification and Validation

Testing and Verification in Software Development

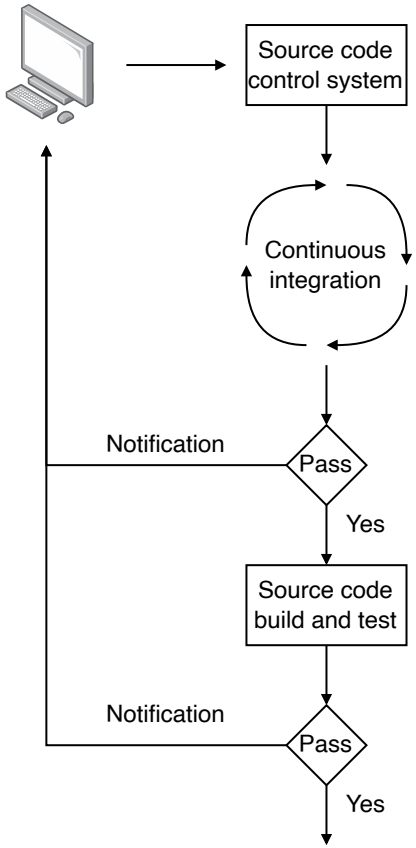
Continuous Integration (CI)

Objective

- Incorporate new and changed code into project code repository
- Find and resolve problems asap
- Ensure system still performs as intended after the new code is integrated

Process

Skip to main content



Continuous integration

Components

- Source code control system
- Build tools
- Test tools
- Scheduler/trigger

Skip to main content

Pros and Cons

Pros

- Early warning of wrong code
- Problems fixed as they occur $\rightarrow \downarrow$ cost of change
- Immediate feedback
- Frequent unit testing
- Can be reverted to last stable version

Cons

- Require setup time
- Cost of procuring machines
- Require time to automate system

Test-Driven Development (TDD)/Test-First Development (TFD)

Philosopy = tests should be written before code

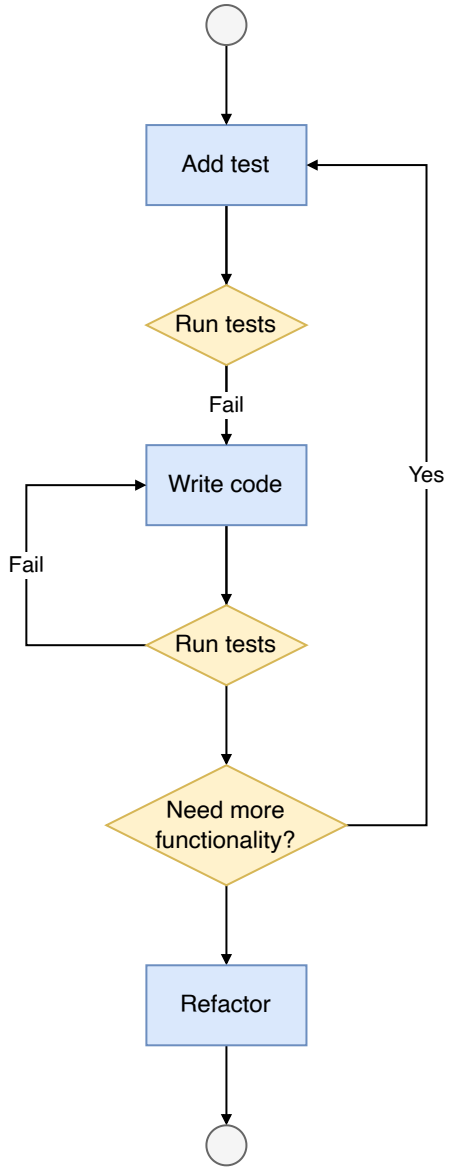
First tests will fail as no code has been written

Start coding $\rightarrow$ run tests until the code passes all tests

Refactoring = clean up design to make it easier to understand and maintain without changing the code's behavior

Red, Green, Refactor/Clean = Writing a test that initially fails, adding code until it passes, and refactoring the code

Skip to main content



TDD process

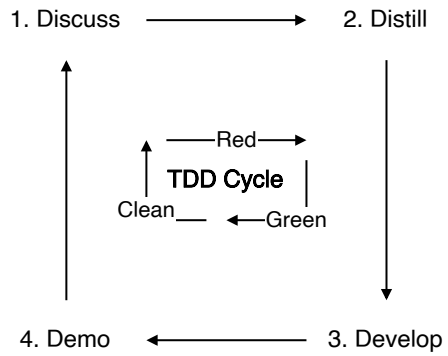
Skip to main content

Acceptance Test-Driven Development (ATDD)

Testing focus on ~~code~~ business

Acceptance tests captured in functional test framework (FIT/FitNesse = Framework Integrated Testing)

Process



ATDD process

1. **Discuss** the requirements = gather acceptance criteria
2. **Distill** tests in a framework-friendly format = structure tests in a table format
3. **Develop** the code and hookup the tests = hook tests to the code and run acceptance tests
4. **Demo** = exploratory testings

Skip to main content

Stakeholder Engagement

Taking Care of Stakeholders

Keeping Stakeholders Engaged

Benefits

- Short iterations keep stakeholders interested in the process
- Hear about change requests as soon as possible
- Identify potential risks, defects, and issues
- Use emotional intelligence and interpersonal skills to understand stakeholders' concerns and find a positive way to engage them with the project
- Establish a process for escalating issues that need a high level of authority to resolve

Incorporating Stakeholder Values

Agile methods focus on bringing project priorities into alignment with stakeholder priorities by

- engaging the PO in the prioritization of the backlog, and
- inviting stakeholders to planning meetings and retrospectives

Skip to main content

Incorporating Community Values

Values shared by Scrum and XP:

- **Respect** = seek consensus
  - Don’t judge suggestions
  - No idea is stupid
- **Courage**
  - Perform early evaluations
  - Focus on transparency by showing
    - Velocity data
    - Defect rates

Establishing a Shared Vision

Agile Chartering

Project charter content:

- Goal
- Purpose
- Composition
- Approach
- Authorization from sponsor to proceed

Skip to main content

Agile versus Non-Agile Charters

Shared goal =

- gain agreement about Who, What, Where, When, Why, How
- obtain authority to proceed

Differences:

- Agile.details < Non-agile.details
- Agile.focus @ How

Developing an Agile Charter

W5H

- **Who**
  - Participants
  - Stakeholders
- **What**
  - Vision
  - Mission
  - Goals
  - Objectives
- **Where**
  - Work sites

Skip to main content

    - ...
- **When**
    - Start
    - End
- **Why** = Business rationale
- **How** = Approach

Project elevator statement

Statement	Description
For	Target customers
Who	Need
The	Product/service name
Is a	Product category
That	Key benefits/reason to buy
Unlike	Primary competitive alternative(s)

Definition of “Done”

- Necessary at all levels (i.e., Deliverables, Releases, and User Stories)
- Consist of multiple **acceptance criteria**

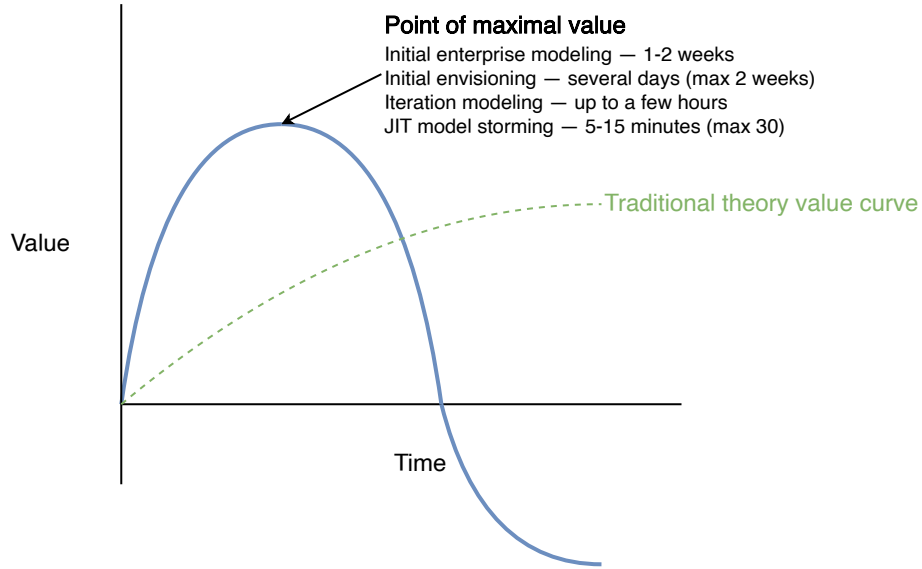
Skip to main content

Agile Modeling

Rationale: value of agile models is in ~~preservation~~ creation

Aim:

- Reflective purposes
- Investigate problems and find solutions



Modeling Value against Time

Types:

- Use case diagrams
- Data models

Skip to main content

Wireframes

Wireframes = quick and cheap mock-up of a product/service

Personas

Personas = quick guides/reminders of key stakeholders and their interests

Augment requirements:

- Help prioritize work
- Stay focused on users
- Gain insights into who users will be

Help empathize with final users of product/service

Keep focus on delivering features that users will find valuable

Communicating with Stakeholders

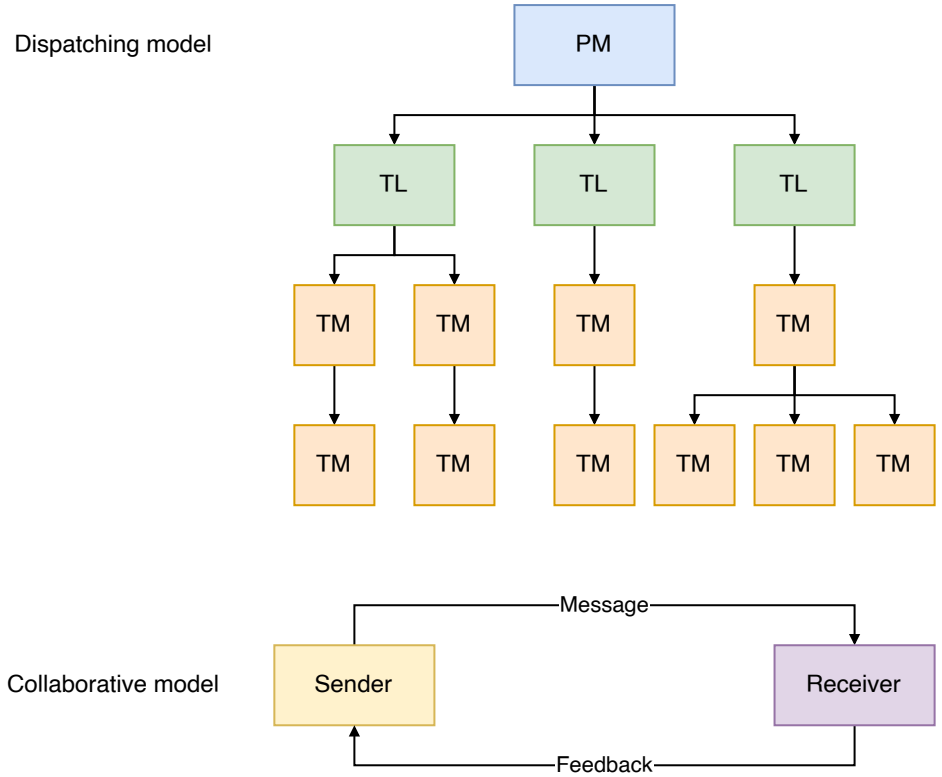
Face-to-Face (F2F) Communication

Highest efficiency: highest interactivity & highest bandwith/information density

Skip to main content

Two-Way Communication

Information flow between stakeholders = **bidirectional**



Dispatching vs Collaborative Communication Model

Knowledge Sharing

Agile projects are encouraged to take an abundance-based—rather than scarcity-based—attitude toward sharing knowledge

Skip to main content

- ↑ #people who know about something, ↑ #people there will be who can help you when you get stuck
- Helps balance the workload between team members

Information Radiators

Information radiators = highly visible displays of information, including:

- large charts
- graphs
- summaries

Data:

- Features delivered vs features remaining
- Who is working on what
- Features selected for the current iteration
- Velocity and defect metrics
- Retrospective findings
- List of threats and issues
- Story maps
- Burn charts

Social Media

Skip to main content

Working Collaboratively

Benefits:

- Generates wiser decisions
- Problem solving
- Fosters action
- Build social capital
- Fosters ownership of collective problems

Workshops

Tips:

- Diverse groups reflect a wider range of viewpoints than just a few experts
- Use techniques to prevent dominant individuals and extroverts from monopolizing the discussion
- Start with an activity that gets everyone participating within the first five minutes

Brainstorming

Methods

Skip to main content

Method	Description
Quiet Writing	5-7 minutes to generate list of ideas
Round-Robin	Take turns to suggest ideas
Free-for-All	Shout ideas

Collaboration Games

Game	Summary
Remember the Future	Vision-setting and requirements-elicitation exercise
Prune the Product Tree	Helps stakeholders gather and shape requirements
Speedboat (aka Sailboat)	Identify threats and opportunities (risks)
Buy a Feature	Prioritization exercise
Bang-for-the-Buck	Value versus cost rankings

Remember the Future

Prune the Product Tree

Skip to main content

Speedboat

Using Critical Interpersonal Skills

Emotional Intelligence

Active Listening

Facilitation

Negotiation

Conflict Resolution

Participatory Decision Making

Participatory Decision Models

Simple Voting

Skip to main content

Thumbs Up/Down/Sideways

Fist-of-Five Voting

Highstmith's Decision Spectrum

Team Performance

Agile Team Roles

- Development Team/Delivery Team
- Product Owner/Customer/Proxy Customer/Value Management Team/Business Representative
- ScrumMaster/Coach/Team Leader
- Project Sponsor

Skip to main content

Development Team/Delivery Team

Product Owner/Customer/Proxy Customer/Value Management Team/Business Representative

ScrumMaster/Coach/Team Leader

Project Sponsor

Building Agile Teams

Development Team

- Size < 12
- Have complementary skills & generalizing specialists with cross-functional skills rather than experts in one field
- Commited to a common purpose
- Hold themselves mutually accountable -> shared ownership for project outcomes

Characteristics of High-Performing Teams

- Create a shared vision for the team
- Set realistic goals

Skip to main content

- Build a sense of team identity
- Provide strong leadership

Models of Team Development

Shu-Ha-Ri Model of Skill Mastery

Acronym	Description
Shu	Obeying the rules
Ha	Consciously moving away from the rules
Ri	Unconsciously finding an individual path

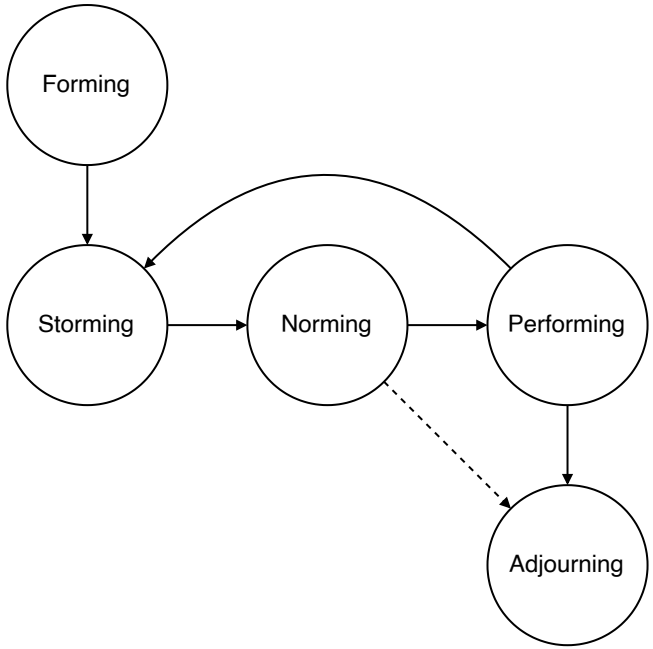
Dreyfus Model of Adult Skill Acquisition

Skip to main content

Stage	Stage	Commitment	Decisions	Perspective
1	Novice	Detached	Analytic	None
2	Advanced beginner	Detached	Analytic	None
3	Competent	Detached understanding and deciding; involved outcome	Analytic	Chosen
4	Proficient	Involved understanding; detached deciding	Analytic	Experienced
5	Expert	Involved	Intuitive	Experienced

Tuckman Model of Team Formation and Development

Skip to main content



Tuckman model team formation and development stages

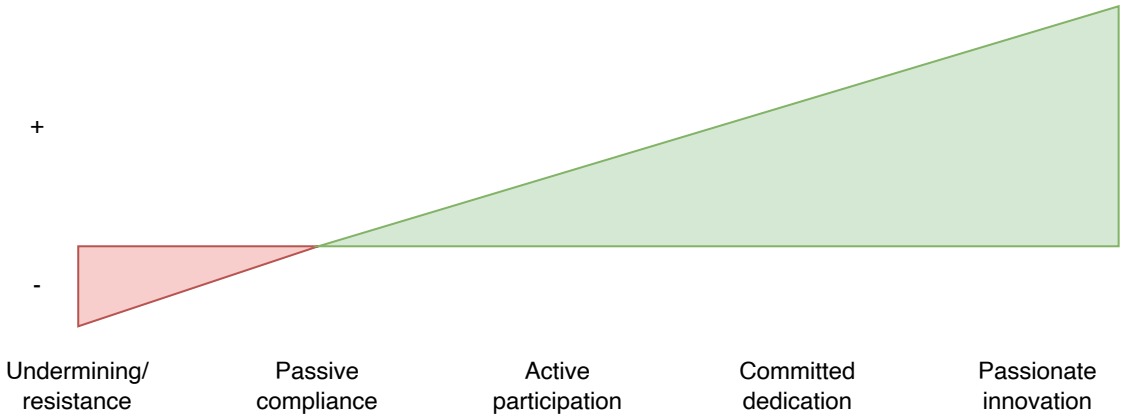
Stage	Description
Forming	Working group
Storming	Pseudo team→ Potential team
Norming	Potential team→ Real team
Performing	Real team→ High performing team

Skip to main content

Adaptive Leadership

Stage	Team Stage	Leadership Style
1	Forming	Directing
2	Storming	Coaching
3	Norming	Supporting
4	Performing	Delegating

Team Motivation



Continuum of Net Contribution

Skip to main content

Training, Coaching, and Mentoring

Training

Training = teaching skills/knowledge through practice and instructions

Coaching

Coaching = facilitated process that helps developing and improving performance

Guidelines for 1-to-1 coaching:

- Meet them a half-step ahead
- Guarantee safety
- Partner with managers
- Create positive regard

Mentoring

Mentoring = professional relantionship where:

- Mentor $\rightarrow$ tackles issues on as an-needed basis
- Mentee $\rightarrow$ free-flowing agenda

Skip to main content

Creating Collaborative Team Spaces

Co-located Teams

Team Space

Osmotic Communication

Global, Cultural, and Team Diversity

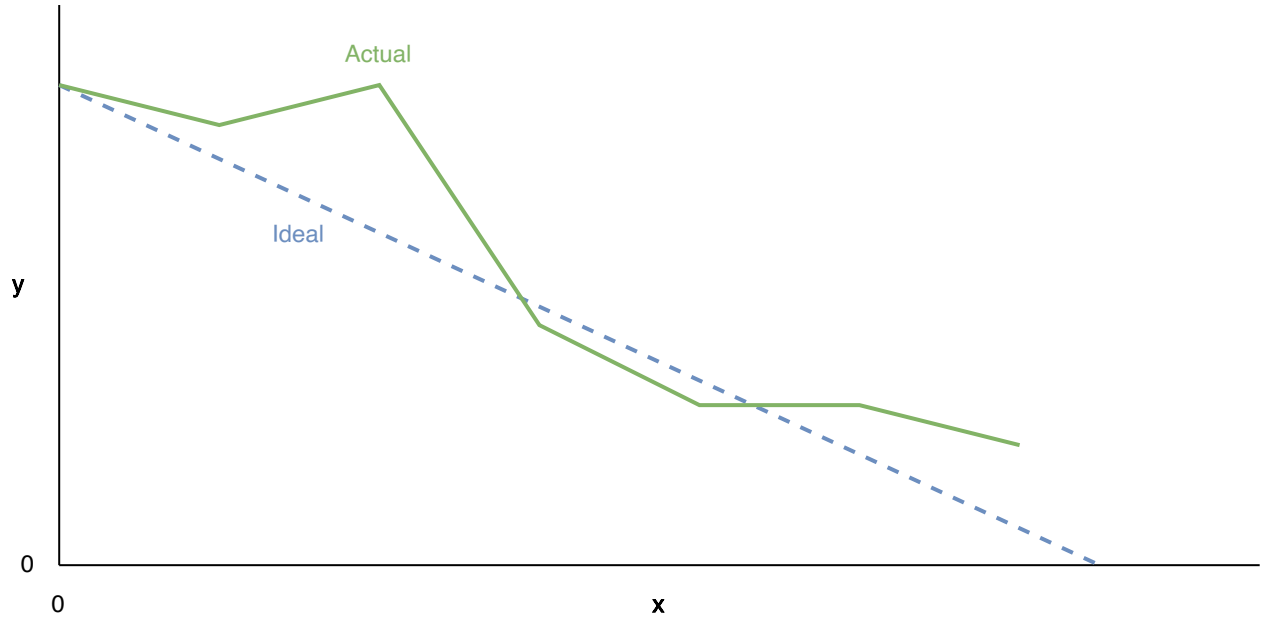
Distributed Teams

Tracking Team Performance

Burn Charts

Burndown Charts

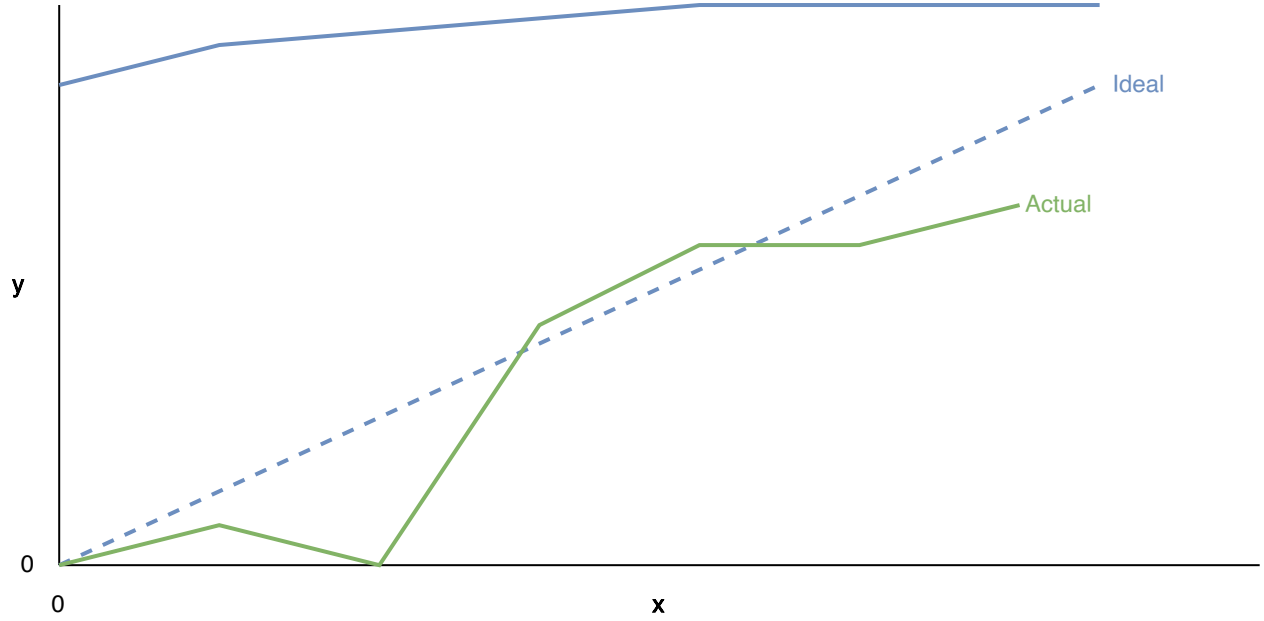
Skip to main content



Burndown chart

Burnup Charts

Skip to main content

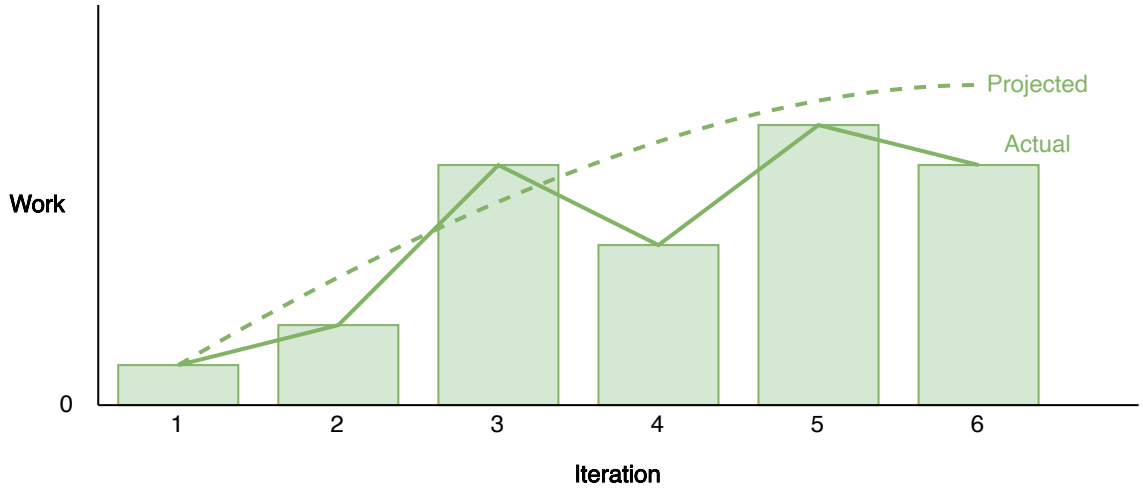


Burnup chart

Velocity

$Velocity = Work/Iteration$, where $Work = StoryPoints, UserStories, Hours, \ldots$

Skip to main content



Velocity chart

Skip to main content

Adaptive Planning

Agile Planning Concepts

Adaptive Planning

Agile versus Non-Agile Planning

Principles of Agile Planning

Agile Discovery

Progressive Elaboration

Value-Based Analysis

Value-Based Decomposition

Timeboxing

Skip to main content

Estimate Ranges

Ideal Time

Ideal Time = Task duration without distractions

Likely Time = Task duration with distractions

Tools for Sizing and Estimating

Sizing, Estimating, and Planning

Decomposition Requirements

Requirements Are Decomposed “Just in Time”

User Stories

Creating the User Stories

Template 1

Skip to main content

Template 2

> Given
>
> When
>
> Then

The Three C’s

C	Description
Card	
Conversation	
Confirmation	

INVEST: Characteristics of Effective User Stories

Skip to main content

Letter	Description
Independent	
Negotiable	
Valuable	
Estimatable	
Small	
Testable	

User Story Backlog (Product Backlog)

Refining (Grooming) the Backlog

Relative Sizing and Story Points

The Fibonacci Sequence

Guidelines for Using Story Points

Skip to main content

- Story point estimates shouldbe all-inclusive
- The point sizes should be relative
- When disaggregating estimates, the totals don’t need to match
- Complexity, work effort, andrisk should all be included in the estimates

Affinity Estimating

T-shirt Sizing

ES < S < M < L < XL < XXL

Story Maps

Product Roadmap

Wideband Delphi

Biases:

- Bandwagon
- HIPPO
- Groupthink

Skip to main content

- Iterative
- Adaptive
- Collaborative

Planning Poker

Release and Iteration Planning

Spikes

Architectural Spike

Short, timeboxed effort dedicated to “proof of concept” — checking whether the approach the team hopes to use will work for the project

Risk-Based Spike

Short, timeboxed effort that makes the team sets aside to investigate, reduce or eliminate an issue/threat

Release Planning

Skip to main content

How Much Can We Get Done?

Estimating Velocity for the First Iteration

Slicing the Stories

Iteration Planning

The Iteration Planning Process

- Discuss the user stories in the backlog
- Select the user stories for the iteration
- Define the acceptance criteria and write the acceptance tests for the stories
- Break down the user stories into tasks
- Estimate the tasks

Iteration Planning Summary

Selecting the User Stories

Defining the Acceptance Criteria nad Writing the Acceptance Tests

Skip to main content

Estimating the Tasks

Use Actual Results to Refine Estimates

Daily Stand-Ups

Problem Detection and Resolution

Detecting Problems

Lead Time and Cycle Time

$Task.LT = Time(ToDo \rightarrow Done)$

Cycle Time, WIP, and Throughput

$CT = WIP/TH$

Throughput and Productivity

$TH = Work/Time$

Skip to main content

Defects

$Defect.CT = Time(Occurred \rightarrow Fixed)$

Defect Rates

$DefectRate = \#Defect/Time$

Variance Analysis

Causes of Variation

- Common cause = average day-2-day differences of doing work
- Special cause = greater degree of variance $\leftarrow$ special/new factors

Trend Analysis

Metrics

- Lagging $\rightarrow$ view of past
- Leading $\rightarrow$ view of future/what is occurring now/starting to happen $\rightarrow$ can adapt/replan accordinly

Control Limits

Skip to main content

Managing Threats and Issues

Risk-Adjusted Backlog

Creating the Risk-Adjusted Backlog

$$EVM = Probability \cdot Impact[\$]$$

Risk Severity

$$Severity = Probability \cdot Impact[l/m/h]$$

Risk Burndown Graphs

Solving Problems

Problem Solving as Continuous Improvement

Engage the Team

The Benefits of Team Engagement

Skip to main content

- By asking the team for a solution, we inherit consensus for the proposal
- Engaging the team accesses a broader knowledge base
- Team solutions are practical
- When consulted, people work hard to generate good ideas
- Asking for help shows confidence, not weakness
- Seeking others' ideas models desired behavior

Considerations and Cautions for Engaging the Team

- Involve the team where it can be most helpful
- Solve real problems
- Team cohesion is necessary
- Check in after team or project changes
- Be sure to follow through

Continuous Improvement

Kaizen

Kaizen = process for continuous improvement

Focus on:

Skip to main content

- frequently initiate and implement small, incremental improvements

PDCA Cycle = Plan - Do - Check - Act

Continuous Improvement—Process

Process Tailoring

Process tailor = adapting our implementation of agile to better fit our project environment

Teams new to agile should use their methodology “out-of-the-box” for a few projects before attempting to change it

All techniques and practices in an agile methodology are designed to work in balance with each other

Hybrid Models

Agile-Agile Hybrid: Scrum-XP

Methodology	**Focus**
XP	Technical guidance
Scrum	Project governance

Skip to main content

Implement agile components into linear project execution

Systems Thinking

Understand the systems-level environment for the project

Figure

Process Analysis

Process Analysis = reviewing and diagnosing issues with a team's agile methods

Methodology Anti-Patterns

- One size for all projects
- Intolerant
- Heavy
- Embellished
- Untried
- Used Once

Success Criteria

- Project got shipped

Skip to main content

- Team would work the same way again

Methodology Success Patterns

- Interactive, face-to-face communication is the cheapest channel for exchanging information
- Excess methodology weight is costly
- Larger teams need heavier methodologies
- Projects with greater criticality require greater ceremony
- Feedback and communication reduce the need for intermediate deliverables
- Discipline, skills, and understanding counter process, formality, and coumentation
- Efficiency is expendable in nonbottleneck activities

Value Stream Mapping

Process

1. Identify product/service to be analyzed
2. Create a value stream map of the current process, identifying steps, queues, delays, and information flows
3. Review the map to find delays, waste, and constraints
4. Create a new value stream map of the desired future state of the process, optimized to remove or reduce delays, waste, and constraints
5. Develop a roadmap for creating the optimized state
6. Plan to revisit the process in the future to continually refine and optimize it

Skip to main content

Metrics

Term	Formula
Total cycle time	$TCT = VAT + NVAT$
Value-added time	VAT
Nonvalue-added time	$NVAT$
Process cycle efficiency	$VAT/TCT = VAT/(VAT + NVAT)$

Project Pre-Mortems

1. Imagine the Failure
2. Generate the Reasons for Failure
3. Consolidate the List
4. Revisit the Plan

Continuous Improvement—Product

Reviews

The Scientific Method

Skip to main content

Product Feedback Loops and Learning Cycles

Feedback Methods

Approved Iterations

Continuous Improvement—People

Retrospectives

Benefits

- Improved productivity
- Improved capability
- Improved quality
- Improved capacity

Process

Skip to main content

Stage	Name	Typical Time
1	Set stage	6
2	Gather data	40
3	Generate insights	25
4	Decide what to do	20
5	Close retrospective	20

Set stage

Activities

- Check-In
- Focus On/Off
  - Inquiry rather than Advocacy
  - Dialogue rather than Debate
  - Conversation rather than Argument
  - Udnerstanding rather than Defending
- ESVP
  - Explorers
  - Shoppers

Skip to main content

    - Prisoners
- Working Agreements

Gather data

Techniques

- Timeline
- Triple Nickels
- Color Code Dots
- Mad, Sad, Glad
- Locate Strengths
- Satisfcation Histogram
- Team Radar
- Like to Like

Generate insights

Five Whys

Fishbone Analysis

Skip to main content

Close retrospective

Team Self-Assessments

Shore’s Team Self-Assessment Scoring Model

Tabaka’s Team Self-Assessment Model

Previous
Design Structure Matrix (DSM)

Next
Scrum Body of Knowledge