

Build Your Agile Business

The Sourcebook



Agile Teams

At this level, empowered, self-organizing, self-managing teams deliver valuable, fully tested software increments every two weeks.

The Daily Meeting	4
Iteration Planning	5
Iteration Planning Agenda	6
User Stories	8
Story Sizing	10
20 Ways to Split User Stories	11
Patterns for Splitting User Stories	12
Defining Done	14

Agile Lessons Learned

Focus on Value

“The benefits of agile are multidimensional. But the most important change is that it focuses the entire organization on meaningful delivery to the customer. Instead of relying on indirect metrics describing degrees of our software’s quality and progress ... we ask, ‘What percent of critical functionality is included in this release?’ This emphasis on customer-perceived value impacts our entire project life cycle pretty significantly.”

Vice President of Infrastructure Management

BMC Software

Agile Teams

The Daily Meeting

The daily standup meeting is for and about the team and its commitments. In this meeting, the team checks in on how their work is progressing in the iteration, adjusts plans and gets assistance with removing impediments.

Who?

- Delivery team
- Product owner
- Scrum Master
- Stakeholders can observe, but are not allowed to speak.
- This is not a status meeting for the product owner.

Timing

Every day, same time, same place, same people. This puts a regular rhythm and cadence on everyone's calendar. The meeting lasts no more than 15 minutes.

Preparation

Individuals collect their achievements and impediments from the previous day. Remember to keep the burndown chart available for reference.

Process

Always stand to keep the meeting short. Only the team speaks. Each team member addresses these questions in a round-robin format:

- What did I do yesterday?
- What am I committing to work on today?
- What is getting in the way of my commitments?
- What new information have I brought since yesterday?

Results

As a result of the daily standup, individuals adjust their daily plans and get help removing impediments. No problem solving occurs in the daily standup. If problem solving must occur, the team ends the standup and starts a different meeting for this work to begin.

Agile Teams

Iteration Planning

The purpose of the iteration planning meeting is for the team to commit to the completion of a set of the highest-ranked product backlog items. These commitments define the iteration backlog.

Before we begin

The team has sized items in the product backlog and assigned relative story point values.

The product backlog has been stack-ranked to reflect the priorities of the product owner.

There is a general understanding of the acceptance criteria for these ranked backlog items.

Determine capacity

The capacity for the team is derived from three measures for each team member:

1. Number of ideal hours in the workday.
2. Days in the iteration that each person will be available.
3. Percentage of time each person will dedicate to this team.

The planning steps

1. The product owner describes the backlog item.
2. The team determines completion tasks for that item.
3. Team members volunteer to own the tasks.
4. Task owners estimate the ideal hours they need to finish their task.
5. Planning continues until the team reaches capacity.

If an individual exceeds their capacity during planning, the team collaborates to better distribute the load.

Agile Teams

Iteration Planning Agenda

1. Opening	Scrum Master
2. Product vision and roadmap	Scrum Master
3. Development status, state of our architecture, results of previous iterations	Product Owner
4. Iteration name and theme	Agile Team
5. Velocity in previous iteration(s)	Scrum Master
6. Iteration timebox (dates, working days)	Scrum Master
7. Team capacity (availability)	Scrum Master
8. Issues and concerns	Agile Team
9. Review and update definition of done	Scrum Master
10. Stories/items from the product backlog to consider	Agile Team
11. Tasking out: tasks, estimates, owners	Product Owner
12. New issues and concerns	Agile Team
13. Dependencies and assumptions	Scrum Master
14. Commit	Scrum Master
15. Communication/logistics plan	Agile Team
16. Parking lot	Scrum Master
17. Action items/plan	Scrum Master
18. Retrospect the meeting	Scrum Master
Close—Celebrate a successful planning meeting	Agile Team

Agile Lessons Learned

Reward Collaboration

“At the most fundamental level, going to agile is very rewarding to a team because the collaboration and the work they do together ultimately creates a product higher in quality with the correct functionality.”

Vice President of Development
McKesson



Agile Teams

User Stories

User stories are value-focused units of delivery that are typically used in agile projects. Written from the customer or stakeholder perspective, user stories share what is needed and why.

Who?

The *who* in a user story is typically someone with a particular role or title, or it could be from the perspective of a persona: a fictitious user's behaviors and needs spelled out in detail.

What?

The *what* in a user story specifies the need, feature or functionality that is desired by the *who*. This is what the team will build into the software or service.

Why?

The *why* in a user story specifies the value, keeping the needs of users and customers front and center.

The Template

The user story template is designed to help product owners and others tell stories with a clear *who*, *what* and *why*:

"As a registered user, I want to reset my password so that I can get back into the site if I forget my password."

"As an unregistered user, I can sign up for the site so that I'm able to have a personalized experience."

"As Tom, I want to only see updates from close friends so that I can view relevant updates during my time online."

Agile Lessons Learned

Get Coaching

“Make sure you get education. Make sure you get people to attend training and get adopted into the agile environment so that they’re ready to hit the floor running when they start.”

Agile Coach

Avaya



Agile Teams

Story Sizing

Agile uses points to size stories. These estimates are relative and are important inputs for good iteration and release planning.

Why size stories?

Story sizing is a way of gathering team insights about what they notice about items going into the product backlog.

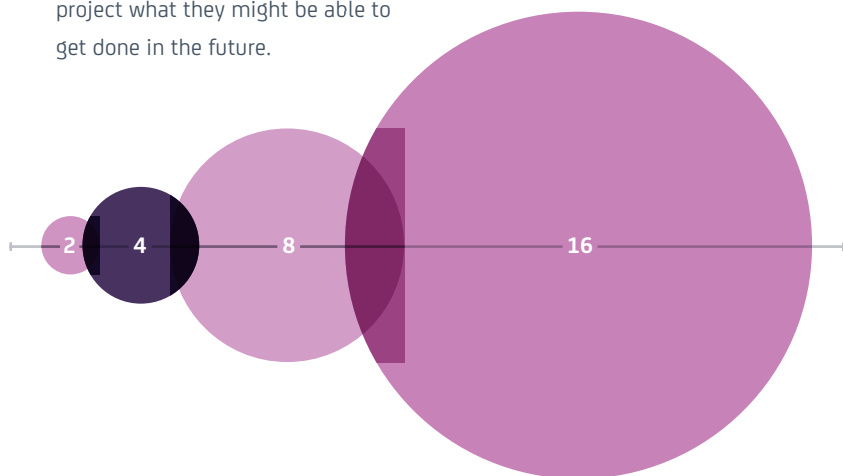
Story sizes:

- Inform the owner of the product backlog how they might move the priority of the item up and down in the backlog.
- Set teams up for continuous improvement for how they plan and project what they might be able to get done in the future.

Why use estimates?

Humans are good at comparing size, but are not very good at estimating absolute numbers. Can you tell the difference between a 1 and a 2? How about between a 33 and a 34?

- Relative size estimates don't change
- Estimating is faster
- Easier to reach accurate consensus on size
- Basic math still works $3 + 3 = 6$



Helpful tip: Estimate by analogy

“Richard’s story is like Anne’s story, so let’s estimate Richard’s story to be the same size as Anne’s.”

Agile Teams

22 Ways to Split User Stories

by Bill Wake

Easier	Harder	Why?
The Big Picture		
Research	Implement	What have others done?
Spike	Implement	Explore a quick solution
Manual	Automated	Often have to retain manual solution anyway
Buy	Build	Can go either way: trade cost of customizing...
Build	Buy	... versus cost of implementing yourself
Single-User	Multi-User	Fewer worries about scale, user accounts
API Only	Interface	Test may function without user interfaces
Character or Script UI	GUI	Simple interface can prove out ideas
Generic UI	Custom UI	"Naked Objects" approach can be cheaper
-ilities		
Static	Dynamic	Do once and ignore updates
Ignore Errors	Handle Errors	Minimize error code (don't ignore exceptions)
Transient	Persistent	Focus on behavior over persistence
Low Fidelity	High Fidelity	Quality of result (e.g., pixel depth)
Unreliable	Reliable	"Perfect uptime is very expensive." Wm. Pietri
Small Scale	Large Scale	Build load capacity over time
Less "ilities"	More "ilities"	Address non-fictional requirements later
Features		
Few Features	Many Features	Easier to do fewer features
Main Flow	Alternative Flows	Happy path vs. all possible paths
0	1	Nothing is easier than something
1	Many	One is easier than a bunch
One Level	All Levels	One level is the base case for all levels
Base Case	General Case	Base case must be done: other needn't

Agile Teams

Patterns for Splitting User Stories

Pattern

Workflow Steps

As a content manager, I can publish a news story to the corporate website.

Business Rule Variations

As a user, I can search for flights with flexible dates.

Major Effort

As a user, I can pay for my flight with VISA, MasterCard, Diners Club, or American Express.

Simple/Complex

As a user, I can search for flights between two destinations.

Variations in Data

As a content manager, I can create news stories.

Data Entry Methods

As a user, I can search for flights between two destinations.

Defer Performance

As a user, I can search for flights between two destinations.

Operations (e.g. CRUD)

As a user, I can manage my account.

Break Out a Spike

As a user, I can pay by credit card.

Example

I can publish a news story

... directly to the corporate website.

... with editor review/with legal review.

... as "n days between x and y."

... as "a weekend in December."

... as " $\pm n$ days of x and y."

I can pay with

... one credit card type (VISA, MC, DC, AMEX).

... all four credit card types (VISA, MC, DC, AMEX).

... specifying a max number of stops.

... including nearby airports.

... using flexible dates.

... in English.

... in Japanese.

... in Arabic.

... using simple date input.

... with a fancy calendar UI.

... slow (just get it done, show a "searching" animation).

... in under 5 seconds.

... sign up or cancel my account.

... edit my account settings.

... investigate credit card processing.

... implement credit card processing (as one or more stories).

Agile Lessons Learned

Product Out the Door

“We always had a working product running, and that saved us a great deal of time. When time started getting close, we could just add onto that to deliver other features as they were needed.”

Project Manager

United States Census



Agile Teams

Defining Done

Agile teams make an agreement about what constitutes potentially shippable software, called the definition of done. It serves as a contract the delivery team writes with its stakeholders, as well as the team's standard of excellence.

How to create a definition of done

1. Write down all of the work necessary for a release. Write each item on a separate Post-it note.
2. Draw three areas on your whiteboard to represent done for a user story, done for an iteration and done for a release.
3. For each item of work, place it in the appropriate section by considering if the team can deliver the work within the boundary.

Work that can't be completed with the story

4. Discuss what obstacles prevent the team from delivering the work with the story.
5. Think creatively about how you might incrementally address impediments.
6. If the obstacle cannot be removed or incrementally addressed, capture it as part of a prioritized backlog of impediments.

Commit and post

7. Call for a fist of five.
8. Prominently post the definition of done to serve as a reminder of the team's commitments to each other.
9. Continue to revisit the definition of done, practicing Kaizen (improvement) and continuously improving your team's operational excellence by moving work away from release boundaries.

Agile Lessons Learned

Share Knowledge

“A culture shift was necessary to our survival. Sure, it takes time. But, if you invest a couple of sprints into a team and get them up to speed on a product area, they can forever share that knowledge and work on that product.”

Chief Agile Methodologist

GXS



Agile Programs

At this level, self-organizing, self-managing teams of agile teams commit to continuous value delivery. They organize around enterprise value streams and align to a common mission.

Scaled Agile Framework®	17
Program Level Definitions	18
Agile Release Train	19
Multiteam Release Planning Agenda	20
Program Level Roles	22

Agile Programs

About the Scaled Agile Framework

The Scaled Agile Framework is a proven knowledge base for implementing agile practices at enterprise scale. Its primary user interface is a big-picture graphic, which highlights the individual roles, teams, activities and artifacts necessary to scale agile from the team, to teams of teams, to the enterprise level.

For more information, please visit scaledagileframework.com

Agile Programs

Program Level Definitions

Value Stream

A sequence of activities intended to produce a consistent set of deliverables of value to customers. This strategy creates the greatest economic benefit to customers.

Feature

At a level higher than stories, Features are services provided by the system that fulfill one or more stakeholder needs. They are maintained in the program backlog and are sized to fit in PSI/releases.

Program Backlog

The single, definitive repository for all of the upcoming work anticipated to advance the Agile Release Train solution. It contains all of the work under consideration.

Potentially Shippable Increment (PSI)

A PSI reflects a milestone, including integration and testing of software assets, across all teams in a Program. It represents a development timebox that facilitates planning, portfolio-level consideration and roadmapping.

PSI/Release

Release whenever you like. Organizations can decide to release more or less frequently than every PSI, or simply release at designated PSI boundaries.

Architectural Runway

Architectural runway exists when the enterprise's platforms have sufficient technological infrastructure to support the implementation of the highest priority business epics in the portfolio backlog without excessive, delay-inducing redesign.

Agile Programs

The Agile Release Train

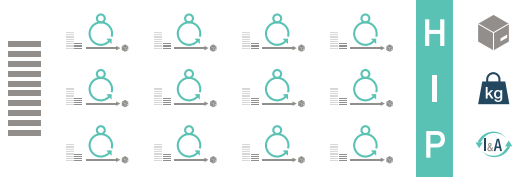
Develop on cadence, deliver on demand.

The Agile Release Train provides a continuing series of incremental releases of value in a value stream—a flow of value in which optimized development and delivery strategy create the greatest economic benefit.

Dates are fixed.

Quality is fixed.

Scope is variable.



Agile Release Train Principles

1. The train departs the station and arrives at the next destination on a reliable schedule.
2. It delivers its cargo to customers (Release) or is used for internal evaluation and proof of incremental system robustness and quality (PSI/Release).
3. If an Agile Team wants its cargo (code, documentation, etc.) to go, it has to put it there on time. The train will not come to them nor wait to get their content.

Train size: Best with 50–100 people

Backlog: Includes features and nonfunctional requirements

Timebox: Scaling up to an 8–10 week timeframe

Team: Includes new roles (see Program Roles page)

Agile Programs

Multiteam Release Planning Agenda

Successful delivery includes:

- ✓ A set of “SMART” objectives for each individual team.
- ✓ A PSI Plan, which highlights the new features, anticipated delivery dates and any other relevant milestones.
- ✓ A vote of confidence/commitment from the entire program to these objectives.

Agenda: Day 1

8:00–9:00	Business context	Senior executive(s)
9:00–10:30	Product/solution vision	Product manager(s)
10:30–11:30	Architecture vision and development practices	CTO, enterprise architect or systems architect
11:30–1:00	Planning context and lunch	Release Train Engineer
1:00–4:00	Team breakouts	Teams
4:00–5:00	Draft plan review	Teams, business owners, stakeholders
5:00–6:00	Management review and problem solving	Management

Agenda: Day 2

8:00–9:00	Planning adjustments	Management team
9:00–11:00	Team breakouts	Teams
11:00–1:00	Final plan review and lunch	Teams
1:00–2:00	Program risks	Teams
2:00–2:15	PSI confidence vote	Teams
2:15–?	Plan rework?	Teams
close	Planning retrospective and moving forward	Release Train Facilitator/facilitator

Agile Lessons Learned

Pick the Right Projects

“A large project carries a certain level of risk, but in turn also encourages inputs from the right people to expose how agile is needed across the rest of the organization. We really wanted a project that people would care about. We needed that level of interest and company investment. Scrum exposes problems that exist in the organization, so we wanted to have all eyes on where we could improve.”

Director, Special Plans and Projects

ChoiceHotels.com



Agile Programs

Program Level Roles



Product manager

Has content authority for the release train and is responsible for defining and prioritizing the program backlog. Works with product owners to optimize feature delivery to customers.



Release train engineer

Acts as the Uber Scrum Master at the program level. Runs the Scrum of Scrums meetings. Coordinates all of the train activities, facilitates program level processes and program execution.



System architect

Has design authority for technological decisions. Understands stakeholder needs and helps teams define and implement a technological solution suitable for hosting current and upcoming features.



System team

Responsible for providing assistance in building and using the development environment infrastructure as well as integrating code from agile teams.



UX designer

Responsible for implementing a consistent user experience across the teams. Works at the program level to provide cross-component and cross-program design guidance.



Release management

Cross-functional team responsible for scheduling, managing and governing synchronized releases (PSIs) across programs or product lines. Coordinates the delivery of software by its associated Agile Release Train.

The Portfolio

At this level, business strategy and development teams align. They collaborate to build realistic roadmaps and prioritize work according to its value and support of the strategic vision.

Strategic Portfolio Planning	25
Portfolio Management	27
Designing a Portfolio Kanban	29

Agile Lessons Learned

Become a Trusted Partner

“We now look at how we can better partner with our customers and make them more successful with our solutions. When we partner, we’re better able to understand our customers’ interests, and ultimately, build long-term relationships.”

DevOps Scrum Master

GE Healthcare



The Portfolio

Strategic Portfolio Planning for Innovation: Six Key Questions

1. Portfolio planning

Where do you want to be in three to five years? Balancing a portfolio begins with a vision. In today's world, a vision points the way and focuses efforts toward value. Determining acceptable investment risks for innovation is a fundamental first step, but many companies lack the visibility they need—across their entire portfolio—to correctly analyze risk, return and value.

2. Financial management

Which investments are you choosing to fund now? Given a three-to-five-year strategy, which investments should you make this year that will incrementally deliver toward your strategy? Carefully selecting where to invest—and especially where not to invest—is challenging but critical to having enough funding left for innovation.

3. Capacity management

Do you have the resources you need for these investments? Do you have the people, the technical expertise and the tools you need to execute on your investments? If not, what's your plan to get them?

4. Value management

What value do you expect to get from your investments? When conditions change, do you re-evaluate the expected return on your investment? A clear understanding of desired outcomes puts you in the driver's seat. Will you speed up, slow down or hit the brakes?

5. Work management

What work do you currently have in flight? Understanding what work is in progress and when you anticipate completing it will guide the sequence of future work to minimize wasteful context-switching and frustrating disruption.

6. Portfolio performance

Does past performance inform your portfolio plans? The empirical knowledge of past work represents the best predictability indicator for future work in similar conditions. Understanding your organizational throughput increases the likelihood to create realistic portfolio plans.

Agile Lessons Learned

Make Work Visible

“It is critical to make our release forecast visible to the product owner, so when we add functionality to a release, we understand what we have to do to add this, or what we’ll have to give up to include that functionality. Those decisions are critical to our business, and most importantly, are now determined by a group discussion earlier in the process.”

Project Manager

Market Force



The Portfolio

Transform Portfolio Management With Lean Thinking

The old way

Value management

- Traditional financial models inhibit innovation
- Big budgets create risk-averse behavior
- Delayed benefits realization
- Subjective, emotional prioritization

Capacity management

- Assign specialized FTEs to projects to maximize resource utilization
- Lost knowledge with constantly reformed project teams
- “Precisely wrong” capacity planning with FTE hourly availability

Financial management

- Fixed scope, schedule and budget inhibits innovative thinking
- Long funding cycles
- Cost overruns

Work management

- Large requirements documents
- Risky big-bang releases
- Work breakdown structure (WBS)
- Diverging from changes to strategy

Portfolio planning

- Big-batch approval processes
- Ad-hoc investment funding
- Siloed and infrequent planning cycles

Portfolio performance

- On-time and on-budget metrics
- Late delivery discovered late
- Subjective progress status
- Infrequent and subjective reviews

The new way

Value management

- Economic value models support innovation
- Taking more, smaller risks increases value creation
- Faster, incremental benefits realization
- Objective, explicit prioritization

Capacity management

- Flow work through cross-functional teams to maximize value delivery
- Retain knowledge with persistent teams
- “Roughly right” capacity planning with cross-functional teams

Financial management

- “Highest value scope first” informs accurate cost and schedule projections
- Incremental funding cycles
- Cost controlled with “just enough” scope

Work management

- Just enough features, just in time
- Minimal viable products (MVP)
- Validated learning
- Aligned to evolving strategy

Portfolio planning

- Smaller batch continuous flow
- Investment funding matches strategy
- Collaborative and iterative prioritization

Portfolio performance

- Customer value metrics
- Fact-based information from real time work progress
- Demonstrable status of high-quality increments of value
- Frequent and objective feedback

Agile Lessons Learned

Be Flexible

“Drop what you are doing, make time and be prepared to be impressed with the results delivered by an expert team of professionals as they conduct their planning session. Plan on leaving energized and filled with ideas and enthusiasm to take back to your organization.”

Director of Project Management

Leica

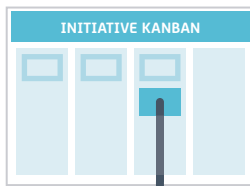


The Portfolio

How to Design a Portfolio Kanban

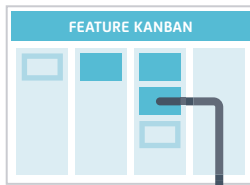
Agile organizations can use Kanban systems at many different levels. For example, an initiative Kanban can feed a feature Kanban, feeding a team's story-level Kanban.

Here are a few options:



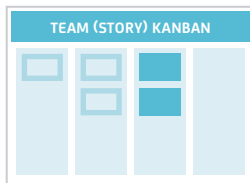
A Kanban of initiatives or projects.

Recommended for teams new to portfolio Kanban. An initiative is a set of work with clear success criteria that one or more teams can complete in a few months.



A Kanban of features or enhancements.

Recommended if you're releasing frequently. This gives you an additional level of Kanban below the initiative.



A Kanban of implementation, content or professional services projects.

Recommended to help you accelerate the speed at which content or studies are completed and reduce coordination overhead.

Get more guidance for building your agile business at ca.com/agile



Connect with CA Technologies at ca.com



CA Technologies (NASDAQ: CA) creates software that fuels transformation for companies and enables them to seize the opportunities of the application economy. Software is at the heart of every business, in every industry. From planning to development to management and security, CA is working with companies worldwide to change the way we live, transact and communicate—across mobile, private and public cloud, distributed and mainframe environments. Learn more at ca.com.

¹ Bill Wake, "Twenty Ways to Split Stories," <http://xp123.com/xplor/xp0512>

² Richard Lawrence, "Patterns for Splitting User Stories," <http://agileforall.com/resources/how-to-split-a-user-story/>

³ Scaled Agile Framework*, <http://www.scaledagileframework.com/>

