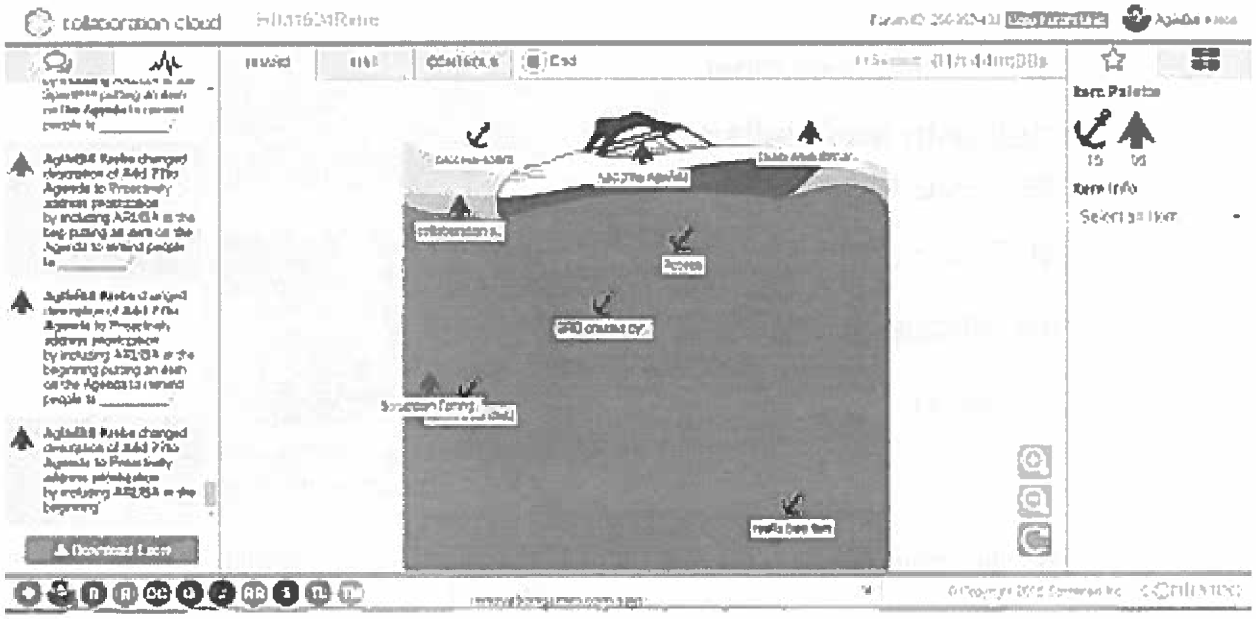
Collaboration Techniques

How can you help the group produce more than just the individuals?

- Find tools that allow group input.
- Can be online, or face to face.



- There may be a wide range of tools and techniques for a given meeting. For example, retrospectives can be supported by Speed Boat, virtual notes as in the Retrium tool, and approaches like dot voting. Many of these have tools to support online, as well as face to face needs.
- Speed boat, shown on the slide, is one example of Collaboration Frameworks. Other formats like *Prune the Product Tree*, or *Buy a Feature*, or *20/20* let you prioritize work. Some, like *Product Box*, let you form high level needs for your products. Some platforms also let you design your own formats.

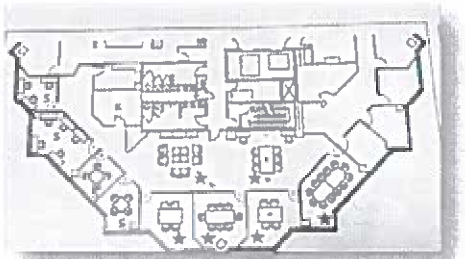
Resources:

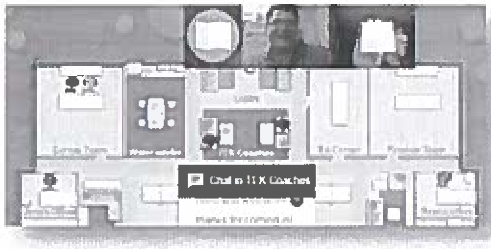
- Tools for distributed retrospectives, prioritization, and analysis: <http://conteneo.co/collaboration-frameworks/>
- Tool for distributed retrospectives: <https://www.retrium.com/>
- Book – *Gamestorming* by Dave Gray and Sunni Brown
- Book – *Innovation Games* by Luke Hohmann
- Book – *Agile Retrospectives* by Derby and Larsen
- Book – *Collaboration Explained* by Jean Tabaka

Discussion: Work Environments and Collaboration

Which environments and communication styles work best?

- Common Room?
  - Information Radiators.
  - People Facing each other.
- Cubicles with low walls; people close together?
- Virtual Co-Location tools?
- Private offices or cubicles?
- Phone calls?


A physical layout


A virtual layout

- High Bandwidth communication is our goal in Agile. Given geographic constraints many companies have, how can we eliminate as many barriers as we can to good dialog?
- **Physical workspace designs.**
- Communication is better when we have:
  - Common Team room (as opposed to individual cubes).
  - Large information radiators (such as burn down charts, task boards, and build progress indicators).
  - Can people be in the same room, same aisle, same floor, or same building? Some Teams find if people have to walk too far, they end up messaging instead.
- **Virtual workspaces**
  - Do virtual Teams need to think about their workspace configuration too? Some teams use electronic tools to categorize topics and messages (such as chat topics in Slack, or Room layouts in Sococo). What working agreements does your team have about how to communicate when?
  - There is a branch of tools that will do better than just screen sharing. Some examples include:
    - Skype: allows you to communicate with distributed Team members and share your screen.
    - Slack: allows you to work efficiently with distributed Team members, categorize topics and messages, and share documents.
    - Sococo: a virtual co-location tool that allows you to set up a virtual office.

This page intentionally left blank.

Assessment: Team Building and Collaboration

Questions:

1. True or False? It is not important for Agile teams to collaborate.
2. True or False? Dysfunctional teams lack commitment and avoid accountability.

Agile Planning: The Vision

Section 8

Levels of Agile Planning

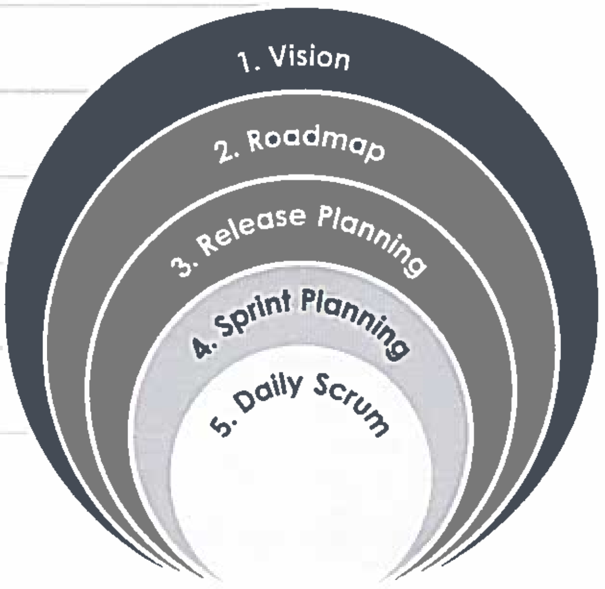
Vision: An opportunity that we have with clients or in the market—a product strategy.

Roadmap: Lays out a high-level plan for when we want to realize this Vision.

Release Planning: Tactical meeting to discuss realistic dates that we can Release the product.

Sprint Planning: The Scrum Team plans to incrementally meet product requirements.

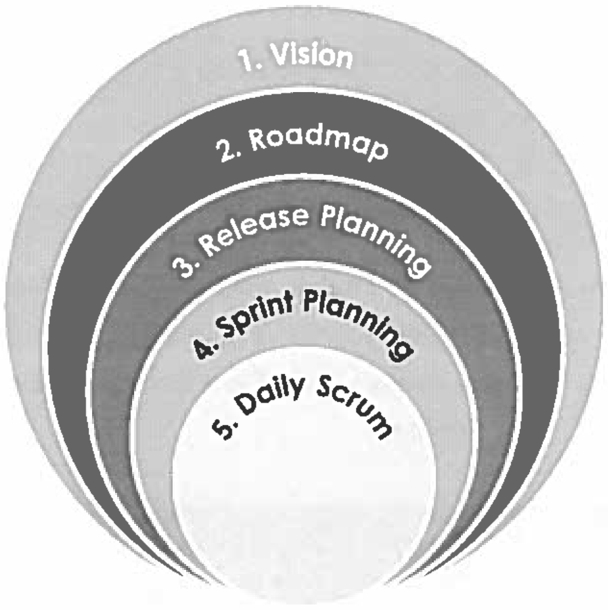
Daily Scrum: Enables the Dev Team to inspect and adapt.



- It is a common myth that Agile methods involve a lack of planning, with Teams diving into Sprints and developing without any upfront planning.
- Regardless of the Agile method, a key input into what happens at the Team level is planning at higher levels in the organization.
- For the purposes of this course and the subsequent simulation exercises, we will be using the more industry-adopted five Levels of Agile Planning.
- The more widely adopted five levels of planning note that Vision and Strategy are the same step; Portfolio Planning is addressed by Roadmap and Release Planning; and Continuous Planning is redundant as a separate step, since planning occurs daily to inspect and adapt.
- Depending on the company or author, Agile Planning has been described as having either five or six levels.

Agile Planning – Product Vision

- Is the overall goal or purpose for the product or service.
- Is a high-level sketch of what the product or service looks like.
- Communicates the essence of the product in a concise manner.
- Reflects common agreement and understanding among stakeholders.



- The Product Vision is the top layer of Agile Planning.
- The Vision paints a picture of the future that draws people in.
- Capturing the Vision is facilitated by the Product Owner or Customer working with any stakeholders, leaders, Subject Matter Experts (SMEs), etc.
- Most importantly, the Customer or Product Owner shares the Vision with the Development Team who will be doing the work to realize that Vision.
- As part of the Product Vision, consider the following:
  - Target: Who are the users or customers?
  - Needs: What problems will this solve? What benefit does it provide?
  - Product: What is the product? What differentiates it from other products?
  - Value: What are the business goals? How does this product benefit the company?

Share the Vision

To share the Vision, the Product Owner will:

- Discuss and Communicate the Vision to:
  - The Development Team.
  - The Stakeholders.
  - The Organization.
- Choose a method to document the Vision:
  - Project Charter or similar document.
  - Business Case, Cost/Benefit Analysis, or Business Model Canvas.
  - Template.

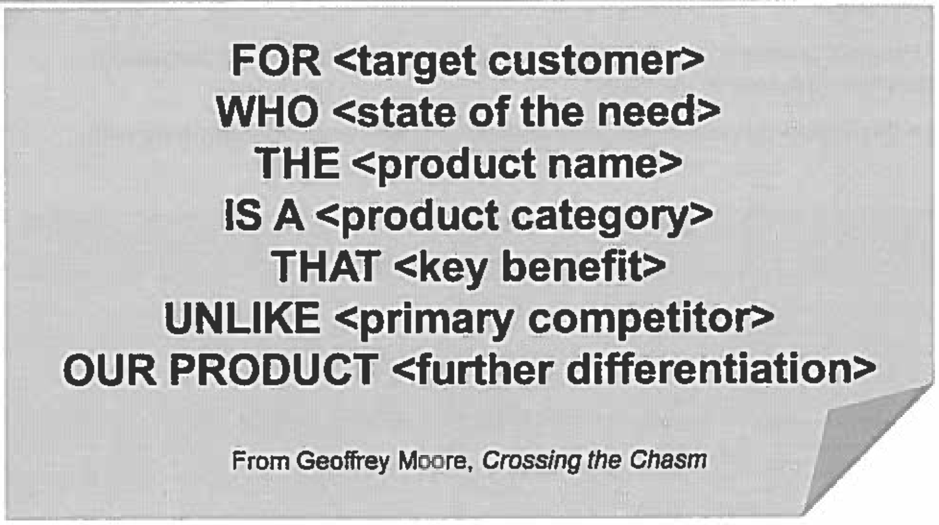
Your Themes, Epics, and User Stories should align with the Vision.

- There is no right or wrong way to develop the product Vision. The important thing to note is that one exists and everyone understands and agrees on what it is.
- The Product Owner is responsible for articulating the Vision. The PO works with key stakeholders, Subject Matter Experts (SMEs), and the Development Team.
- The Development Team should know what the Vision is and where it is. If they don't then the Product Owner has not clearly articulated or shared it. Some Teams have Vision statements printed out and posted visibly on the wall of their Team area, next to task boards.

TEKsystems
Our people make IT possible.

Elevator Pitch Template

A widely adopted template to document the Product Vision is Geoffrey Moore's Elevator Pitch from his book "Crossing the Chasm"



FOR <target customer>
WHO <state of the need>
THE <product name>
IS A <product category>
THAT <key benefit>
UNLIKE <primary competitor>
OUR PRODUCT <further differentiation>

From Geoffrey Moore, *Crossing the Chasm*

- Although there is no standard template, or one way to capture the Product Vision, many in the Agile community have adopted Geoffrey Moore's "elevator pitch" template from his book *Crossing the Chasm.*
- It is brief, to the point and easy for most Product Owners to be able to use to succinctly capture the Vision and share it with the Development Team.
- Its name comes from the fact that someone should be able to communicate this in a few minutes, or the time it takes to ride in an elevator to answer the question "what are you working on?"

Exercise: Vision

Instructions:

1. In your Teams, create a Vision for your product or service using the Geoffrey Moore template.
2. If your Product Owner is in attendance, he or she should drive this discussion with input from the rest of the Team.
3. Capture the Vision on paper and be prepared to share what you came up with.

FOR <target customer>
WHO <state of the need>
THE <product name>
IS A <product category>
THAT <key benefit>.
UNLIKE <primary competitor>
OUR PRODUCT <further differentiation>

From Geoffrey Moore, *Crossing the Chasm*

Discussion: Vision

15 min

Questions:

- In your organization, has someone been responsible for determining the Product or Project Vision? If yes, who is this? If no, what challenges do you see there?
- In order for your Scrum Team or organization to successfully adopt Agile, how important is it to understand the Vision—what you're actually being asked to deliver?

Assessment: Agile Planning: The Vision

Questions:

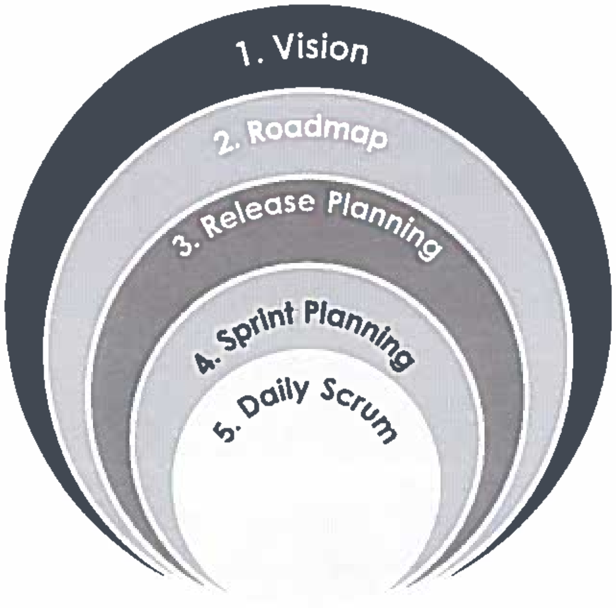
1. True or False? The Development Team discusses and communicates the Vision to the stakeholders and organization.
2. True or False? Geoffrey Moore's Elevator Pitch is a widely adopted template to document the Product Vision.

Agile Planning: The Roadmap

Agile Planning – Product Roadmap

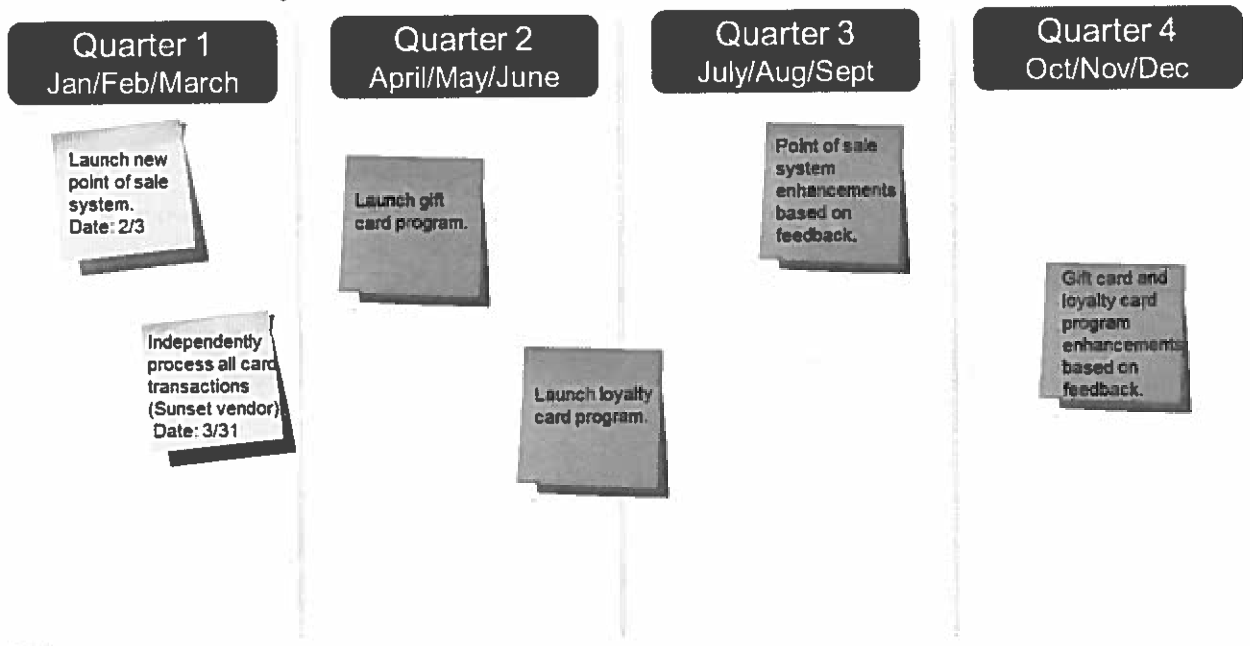
Roadmap Planning:

- Typically takes place after establishing the Vision.
- Ensures priorities are aligned at all levels including the Scrum Teams executing the Vision.
- Is not necessarily a step in Scrum or other Agile methods, but has been widely adopted.
- Is visualized in a Product Roadmap.



- Vision Planning can occur at any time in an organization's fiscal year. Maybe this was an idea previously identified or it's in response to a competing product or service or the result of research, customer focus groups, etc. The Vision or idea then drives the Roadmap.
- At a high level, the organization must decide what its priorities are for projects, initiatives, etc. It is important to note that at the Roadmap level, we have not engaged the people who will be doing the work who can best tell us how long the work will take. For that reason, Roadmap plans are high-level goals and not exact dates. Instead, it is expressed at the quarter level (Q1, Q2, etc.) or perhaps the time of year (e.g., first half of the year, second half of the year).
- The Product Owner is responsible for keeping the Roadmap current and visible to the Team executing it.
- The Product Roadmap provides the link between the Product Vision and the strategy for executing that Vision and it must be communicated to the organization, stakeholders, and the Development Team.
- Keep the Roadmap simple and easy to understand.

Product Roadmap: Evolving Across Releases


Quarter 1
Jan/Feb/March
Quarter 2
April/May/June
Quarter 3
July/Aug/Sept
Quarter 4
Oct/Nov/Dec
Launch new point of sale system. Date: 2/3
Independently process all card transactions (Sunset vendor) Date: 3/31
Launch gift card program.
Launch loyalty card program.
Point of sale system enhancements based on feedback.
Gift card and loyalty card program enhancements based on feedback.

▢ Forecasted plan, dates, and work

▨ Longer term: No detailed requirements, no forecasted dates, etc.
Subject to big changes based on business needs and value

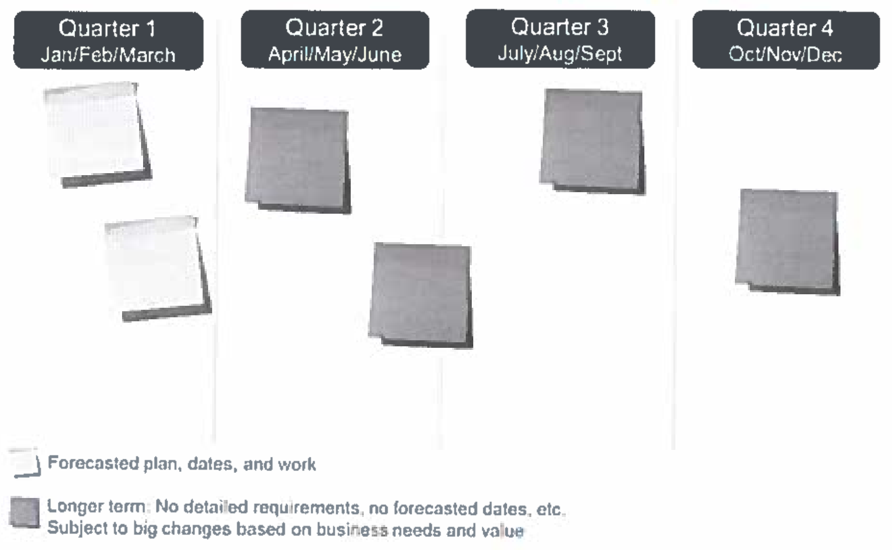
- The Product Roadmap can help an organization or Scrum Team:
  - Show how the product will evolve across Releases or versions.
  - Facilitate dialog between stakeholders, Product Owner, and the Scrum Team.
  - Simply state essential and ordered high-level milestones for the product(s).
  - Focus on the Product Backlog from Vision through execution using the high-level timeframe outlined on the Roadmap.
- The Roadmap should continually be evaluated, shifting as new things come on to the plate, others drop off, etc.
- Product Roadmap Example: On this slide is an example of a Roadmap where a Team or Teams has responsibility to release one, versioned product:
  - The quarter that is yellow is the one closest to us (nearer timeframe), so those initiatives already have budget and are either ready to hold Release Planning or have held Release Planning.
  - The quarters that are in blue are further out and may change based on feedback we get from customers, market reaction, competitors reaction, etc.
  - As those further timelines draw closer, the Product Owner makes decisions on what features to move forward with in the "next version," holds Release Planning, and secures funding to kick off that next "project phase."

Exercise: Roadmap

20 min

Instructions:

- In your Teams, create a high-level Roadmap for your product:
    - Please note that the Vision provides key input into this level of planning.
    - If the Product Owner is in attendance, have them drive this discussion, taking into account the organization's fiscal year or Release cycle.
- Capture the Roadmap on paper and be prepared to share your results.


Quarter 1
Jan/Feb/March
Quarter 2
April/May/June
Quarter 3
July/Aug/Sept
Quarter 4
Oct/Nov/Dec
Forecasted plan, dates, and work
Longer term: No detailed requirements, no forecasted dates, etc.
Subject to big changes based on business needs and value

Assessment: Agile Planning: The Roadmap

Questions:

1. True or False? The Product Roadmap can help an organization show how the product will evolve across Releases.
2. True or False? The Product Roadmap is finalized before the project starts and does not change.

Agile and the Customer

Agile Processes and the Customer

The customer is at the heart of Agile:

- Our highest priority is to satisfy the customer with early and continuous delivery of valuable products or software.
- Agile processes use change as a competitive advantage for the customer.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business owners and developers must work together daily throughout the cycle.

- There can be a misperception about Agile being "for developers" or that it is an "IT process."
- It is actually a business or customer-driven process.
- Without the Product Owner or a customer driving the Vision and the business priority ensuring that we will deliver value with each Iteration, Teams are left to try to guess about what to focus on.
- Agile welcomes changing requirements, even late in development.

Who is the Customer?

The customer is the central focus of Agile and is actively involved throughout the cycle.

- In order to best define the Product Vision, we need to focus on customers:
  - Who are they?
  - What's important to them?
  - How will they use the product or service?
  - What are their priorities?
  - Are there any characteristics we need to consider, such as technical or computer proficiency, or lack thereof?

- The customer is the central focus of Agile and is actively involved throughout.
- A useful exercise that is often overlooked in Agile Planning is spending some time talking about who the end user or the customer for the product or service will be.
- If we understand more about the user, it can flush out important details about the feature or functionality.

User Roles

- With Agile methods, requirements are conveyed by the customer and represent the user's point of view.
- Unique user perspectives help flesh out emerging requirements and help define what we mean by "done."
- Identifying users can be as simple as categorizing the user roles that exist for a product or service:
  - For example, for a point of sale system at a retail store, the user roles may be "cashier," shift leader," and "store manager."

- Identifying users can be as simple as categorizing the user roles that exist for the product or service.
- For example, for a point of sale system at a retail store, some user roles may be cashier, shift leader, and store manager.

What is a Persona?

- An imaginary character who represents a user role.
- Useful when you do not have easy access to real users.
- Creating a persona involves more than coming up with a name for the character:
  - What job do they do for a living?
  - What demographic do they represent?
  - What is their educational background?
- Major characteristics of a *user* or *group of users*.
- A persona can be a *positive* or *negative* character to demonstrate desired or undesired functionality.

- Identifying user roles is a great start. For more important user roles, there's a set of named characteristics that takes this a step further, called personas.
- The persona technique was first introduced by Alan Cooper, to define an archetypical user of a system; an example of the kind of person who would interact with it.
- Personas are used to represent different user types within a targeted demographic identifying the goals and behavior of this hypothesized group of users.
- A persona is a fictional representation of a set of user characteristics; it is not any actual user's name.
- Creating personas is not part of Scrum or Agile, but many find it to be a useful technique in understanding requirements from the user's point of view—which is an idea at the core of any Agile approach.

Example Persona
Connie the Cashier:

- Connie is a cashier in a retail department store.
- She has worked at the store for 5 years.
- Connie is a part-time, evening, and weekend employee.
- She has some computer experience, but mostly for home use in word processing, checking email, and shopping online.
- Connie has a journalism degree and has been submitting writing samples to magazines to try to land a full-time role as a journalist.

- "Connie the Cashier" is an example of a persona.
- The idea is that whenever someone on the Team refers to "Connie the Cashier," these characteristics come to mind and there's an understanding of the level of detail needed in the requirements, or the persona can be used to focus the requirements discussion.
- Discuss with other class participants: Do you find this technique useful? Is it one that you have tried in your Teams?

Example of Negative Persona

Harry the Hacker:

- Harry makes his money by:
  - Setting up online virus scams.
  - Hacking into unsecured databases to obtain Social Security or credit card numbers.
- He has a high degree of computer proficiency and has advanced knowledge of security systems for online applications.
- He works a day job that he would like to quit.

- Personas can also be negative to illustrate a point or to ensure that the Product Owner understands the implications of not prioritizing something that may—at a glance—be considered technical, like the need for security.
- "Harry the Hacker" is an example of a negative persona.
- The idea is that whenever someone on the Team refers to "Harry the Hacker," these characteristics come to mind and there's an understanding of the level of detail needed in the requirements, or the persona can focus the requirements discussion.
- Giving the character a name and talking about his or her less than noble intentions can give that requirement new light, and it can get prioritized more appropriately along with other requirements.

Exercise: User Roles & Personas

15 min

Instructions:

1. In your Teams, identify the product, service, or project you will use for our simulation.
2. Discuss the user roles and personas for this product or service and document any that you come up with.
3. If your Product Owner is in attendance, he or she should drive this discussion with the Team and SMEs providing input.

Assessment: Agile and the Customer

Questions:

1. An imaginary character who represents a user role is called a:
   A. User
   [✓ B.] Persona
   C. Customer
2. [✓ True] or False? With Agile methods, requirements are conveyed by the customer and represent the user's point of view.

User Stories and the Product Backlog

User Stories

Email Details
As a Restaurant Seeker, I want to email reservation details to a list of people so that they know about the dinner reservation.

- A User Story is a requirement or feature expressed *briefly* from the user's perspective.
- Stories are a promise to continue the conversation.
- User Stories describe:
  - Who has a particular need or want.
  - What that need or want is.
  - Why they need it.

- A *User Story* is a requirement or feature expressed briefly from the user's perspective. The few lines of the User Story are not meant to be the requirement in its entirety.
- It is meant to be the start of an evolving conversation.
- User Stories describe who has a particular need or want, what that need or want is, and why they need it:
  - The Why is important! Without it, we have not captured the business value or need and have taken away one of the items that helps show priority.
- In traditional processes, like waterfall, a large document took the place of the conversation.
- Because we value individuals and interactions over processes and tools and working software over comprehensive documentation, we want to collaborate and talk with each other about the Story, documenting the results of the conversation.
- User Stories are not an official part of the Scrum framework, but they have become common in the Agile community.

What is the Product Backlog?

Order

Story #	Story Points	Description
2	3	Stand up dev server for project
1	8	User Story 1
5	5	Fix support ticket #3001
3	3	Research display options
4	5	User Story 2
6	2	Upgrade to Flex version 4
9	8	User Story 3
8	1	User Story 4
7	13	User Story 5
10	?	Fix defect RA2441

- The Product Backlog is the high-level to-do list for the product, and is the responsibility of the Product Owner. It is the single source of requirements for changes to the product. The Product Backlog:
  - Is an ordered list of all the work we have to do to release the product.
  - Includes the features, functions, User Stories, requirements, enhancements, and fixes that constitute changes to be made to the product in future Releases.
  - Can also include foundational work, constraints, or analysis we need to perform.
- Items on the Product Backlog are frequently, but not always expressed as User Stories. Non-functional requirements, support tickets, technical considerations and other items that can't be expressed from the end user's point of view may simply be referenced in plain language.
- Each item in the Product Backlog has a rough estimate of the business value and development effort:
  - The Product Owner orders each item on the Backlog in relation to the other items on the Backlog.
  - Items in the Product Backlog are ordered based on considerations such as risk, business value, dependencies, and date needed. Items higher in the list have more detail, while items further down the priority list are still broadly and imprecisely defined.
- The Product Backlog does not address how the work is completed, only the scope of work that needs to be created.

This page intentionally left blank.

The 3 C's of User Stories

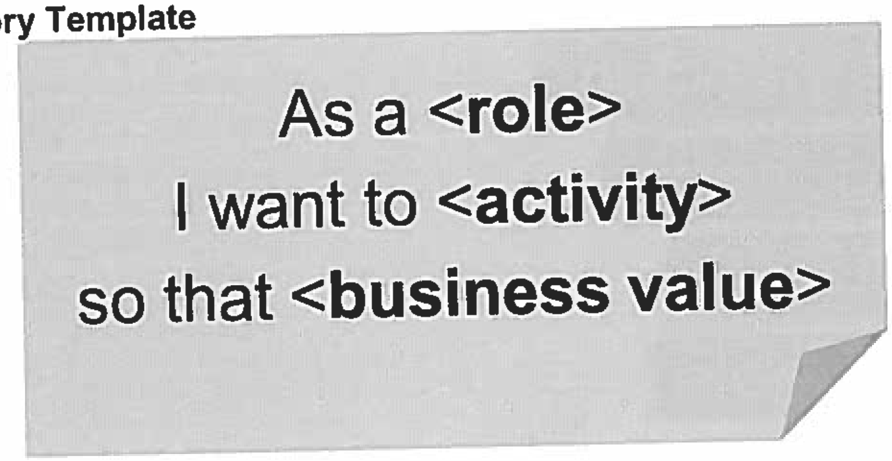
Card	Traditionally captured on a note card; cards may be annotated with estimates, notes etc.	As a user, I want to login and gain access to the www.esCARgot.com website, so that I can purchase a car.
Conversation	Details captured in conversations with Product Owner.	What if my account is expired? Will I remember my login? Can I reset my password?
Confirmation	Acceptance Criteria confirms that the Story was coded correctly.	Acceptance Criteria: 1. Expired accounts fail. 2. It remembers the login, but not the password. 3. I can reset my password.

Source: XP Magazine 8/30/2001, Ron Jeffries

- A User Story is a short description of something your customer will do when they come to your website or use your product or service. It is focused on the value or result they get from doing this thing. In other words, it is a tool and technique used in Agile software development to express requirements. It captures a description of a software feature from the end-user perspective and is short enough to be completed in one Sprint.
- According to Ron Jeffries, there are Three C's of User Stories: Card, Conversation, and Confirmation:
  - Card: The card is a short statement of functionality usually captured on an index card or sticky note for visual organization, and may be managed in software, such as Rally. It describes the type of user, what they want, and why. The statement should use the language of the customer so that it is clear to both the business and the Development Team what the customer wants and why. Physical cards are excellent when eliciting, prioritizing, and organizing User Stories, but the card is a reminder or placeholder for a deeper conversation. A card is not meant to be just the Agile version of a long, detailed written requirement.
  - Conversation: The few lines of a User Story are not meant to be the requirement in its entirety. They are meant to be the start of an evolving and ongoing conversation. A User Story requires collaboration and conversations between business owners and the Development Team to clarify the details as the code is developed. Because we value individuals and interactions over processes and tools and working software over comprehensive documentation, we want to collaborate and talk with each other about the Story, documenting the RESULTS of the conversation, the Acceptance Criteria.

    - Confirmation: User Stories are given more detail as the Conversation is held and things evolve and emerge. These details, called Acceptance Criteria also could be understood as high-level test cases—the cases that must be met for the business to accept a given Story.
- The Development Team determines how to satisfy the requirements of the User Story.

A User Story Template



As a **<role>**

I want to **<activity>**

so that **<business value>**

- **Role** ("Who"): Represents who is performing the action. It should be a single person, not a department. It may be a system if that is what is initiating the activity.
- **Activity** ("What"): Represents the action to be performed by the system.
- **Business Value** ("Why"): Represents the value to the business. "Why is this Story important?"

- The three sections of a User Story are the Role (Who?), the Activity (What?), and the Business Value (Why?).
- The Why is important. Without it, we have not captured the business value or need and have taken away one of the items that helps indicate priority.

INVEST in Good Stories

I Independent

N Negotiable

V Valuable

E Estimable

S Small [Small enough to fit in a Sprint]

T Testable

Source: Bill Wake, http://xp123.com/articles/invest-in-good-Stories-and-smart-tasks/

There are no hard and fast rules for writing User Stories, so how do you tell whether or not a User Story is good? The INVEST acronym is simple way to evaluate User Stories. INVEST is an acronym for the characteristics of a good User Story:

- **Independent.** Ideally the User Story should not be dependent on other Stories. We try to keep the functionality discrete so that we do not set up intricate dependencies that prevent us from getting to "done" by the end of the Sprint. Ask: Is the Story dependent on other Stories?
- **Negotiable.** User Stories are not cast in stone. They are reminders to have a conversation and will evolve as the Product Owner and the Development Team discuss them. User Stories should have enough information to capture the essence of the feature without requiring too much collaboration for the basics, but they should not have too much detail because then the conversations may not happen. Ask: Is there enough information? What other information would I need?
- **Valuable.** User Stories should have value to the user or owner of the solution. If there is no "why" statement that expresses the business value, we may not need to work on this feature. Why work on something that will realize no value? Ask: Is the Story valuable to the user or the owner? What is the value?
- **Estimable.** The Development Team should be able to estimate the User Story. A User Story may not be estimable if it is too vague, doesn't have enough information, or is too big. Ask: Could you estimate this Story? If not, is the Story too big? Too vague? Missing too much information?
- **Small.** User Stories should be small, but not too small. Sometimes this is referred to as right-sized. By the time you include them in a Sprint, User Stories should be small enough that several can be completed in one Sprint, but not so small that they could be considered tasks. Ask: Can the Story be completed in one Sprint? Is the Story too small?

- **Testable.** User Stories should be testable. We must have a way of checking the Story against the Acceptance Criteria so that we know we have achieved our Definition of Done. Ask: Is the Story testable?

User Story Examples

View Details
As a Restaurant Seeker, I want to view special offers of a selected restaurant so that I can find a meal for my taste and budget.

Follow Up
As a Restaurant Seeker, I want to enter my ratings for the restaurant online so that I can share my experience with others and remember my opinion.

Here are some additional User Story examples of the INVEST model at work:

- Even though these Stories are part of a larger application or website, they can be completed independently.
- If someone asks the Product Owner, "What do you mean by ratings – food? Ambiance? Service?" the Product Owner may realize that this is a vague Story so he or she clarifies in the language of the Story or in the Acceptance Criteria as a result of the Negotiation with the Development Team.
- Both Stories are Valuable, as we're told why they are important to the end user or business.
- Both Stories are able to be Estimated, so they are not Epics but are well-sized User Stories.
- Both Stories are Small or Sized right, as they should be able to be completed within a Sprint.

Exercise: Create User Stories

10 min

Overview: Create 3-5 User Stories for the product you chose earlier today. Use your Product Vision and the roles and personas you identified to help.

Instructions:

- Break into your groups.
- Based on your Product Vision and the roles and personas you identified for your product, create 3-5 User Stories. (More is better).
- Be prepared to share with the class.

As a **<role>**

I want to **<activity>**

so that **<business value>**

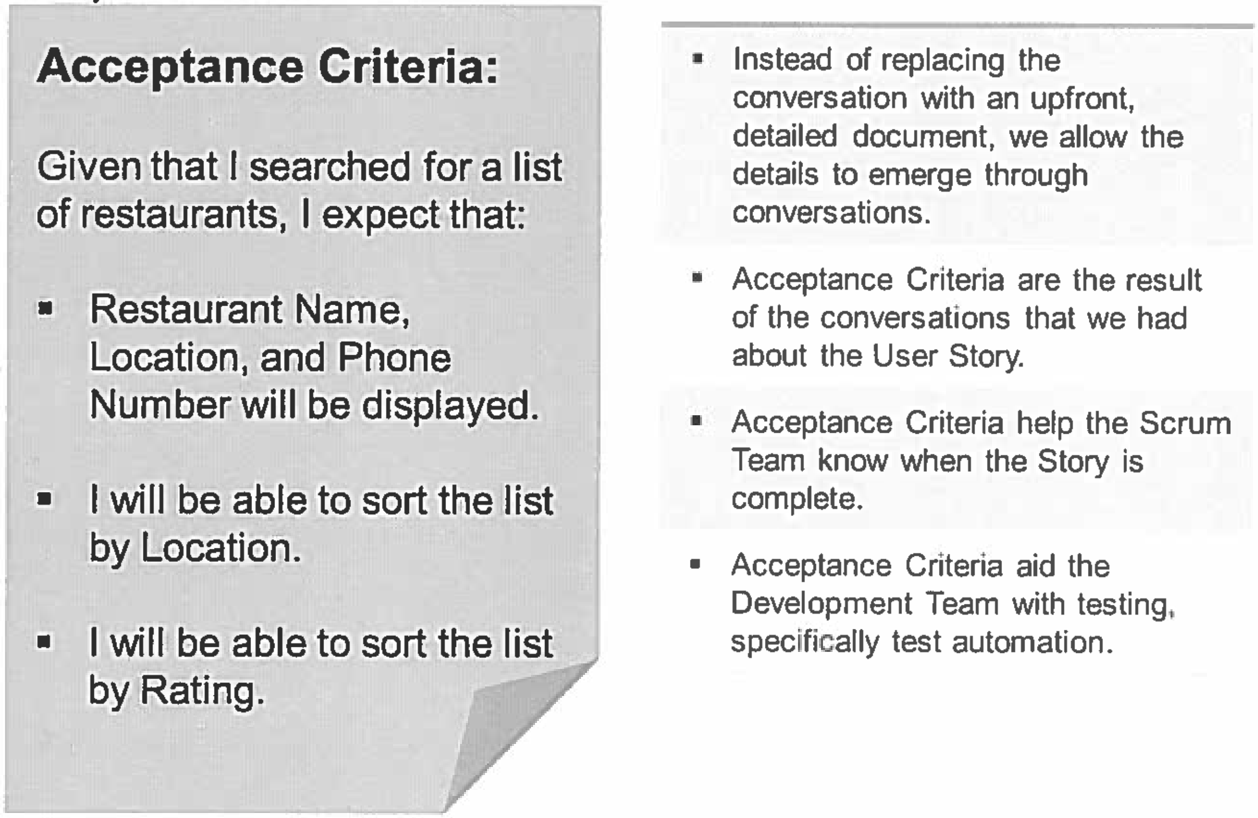
User Story Conversations

User Story Details? Ask the Product Owner.

- Are there other options for landing at this particular page?
- Are we limited by user security or access?
- Do we need to account for error handling or the "sad path" in addition to the "happy path"?
- How do we confirm the functionality works correctly?
- Do you think we have all of the Acceptance Criteria for this Story?

- The initial User Story should fit on an index card or sticky note capturing the Who, What, and Why in a brief description that everyone understands, but it likely doesn't have all the details you need to actually work on the Story.
- That's because Stories are not contracts or set in stone—they are emergent. Details are captured as they unfold. And, they unfold as the Development Team has conversations with the Product Owner.
- The questions that appear on this slide are typical ones the Development Team would ask the Product Owner to clarify the Story.
- Development Team members should capture the answers as Acceptance Criteria. Depending on how the User Story is documented, you might put the Acceptance Criteria on the back of the User Story sticky note or card, in a Word doc, or in a tool where Story information is kept (e.g., Rally).
- The Story should be visible and accessible to the Development Team when they start to work on it.

Acceptance Criteria



Acceptance Criteria:

Given that I searched for a list of restaurants, I expect that:

- Restaurant Name, Location, and Phone Number will be displayed.
- I will be able to sort the list by Location.
- I will be able to sort the list by Rating.

- Instead of replacing the conversation with an upfront, detailed document, we allow the details to emerge through conversations.
- Acceptance Criteria are the result of the conversations that we had about the User Story.
- Acceptance Criteria help the Scrum Team know when the Story is complete.
- Acceptance Criteria aid the Development Team with testing, specifically test automation.

- In traditional processes, we received large documents to read through at the beginning of a project.
- Often times there were other roles to go through as "go betweens" to get the answer from the customer, end user, or voice of the customer.
- Agile strips the back and forth out and has the voice of the customer, the Product Owner, in the driver's seat and accessible to the Development Team.
- Agile also forces the conversation by not having a large document available at the beginning of the project.
- We document the results of the conversation—we do not replace conversation with documentation.
- As conversation takes place, emergent details should be captured.
- These may be valuable Acceptance Criteria that we want to ensure the Development Team who is doing the work understands.
- There is no rule, formula, or prescribed template for how to do this.
- Scrum Teams are empowered to do this in whatever manner works best for them whether a document is leverage, an Agile project management tool, notes written on the back of the initial Story index card, etc.
- The important thing is to have Acceptance Criteria so the Development Team knows.

- They need to define what the Definition of Done is by the end of the Sprint.

Exercise: Create Acceptance Criteria

Overview: For the User Stories you created in the last exercise, determine your Acceptance Criteria.

Instructions:

- Break into your Teams.
- For each User Story, create the Acceptance Criteria.
- Ask yourself: How do you know you're done? How do you know it works? And based on the answers, what are a few things you would determine as Acceptance Criteria.
- Complete criteria for as many Stories as you can.
- Be prepared to share.

Assessment: User Stories and Product Backlog

Questions:

1. What helps the Scrum Team know when the Story is complete?
   - A. The Business Value or "Why" of the User Story
   - B. Acceptance Criteria
   - C. INVEST
2. What are the three Cs of a User Story?
   - A. Card
   - B. Confirmation
   - C. Collaboration
   - D. Conversation
3. Which of these can be items on the Product Backlog?
   - A. User Stories
   - B. Bugs and Defects
   - C. Non-functional Requirements
   - D. All of these

Definition of Done

Section 12

"DoD"

Story Acceptance Criteria vs. Definition of Done

Acceptance Criteria:	Definition of Done:
▪ Describes correct behavior of functionality—proving it works.	▪ Describes a minimum software quality standard.
▪ Applies to only one Story.	▪ Applies to all Stories.
▪ Product Owner has the final say.	▪ Decided/created by the Scrum Team.
▪ Created during refinement sessions.	▪ Created prior to Sprint 1.
	▪ Should eventually represent a "Releasable Increment" standard.
	▪ Continuously improved over time and during Sprint Retrospective.

- As you know, Acceptance Criteria describes the correct behavior of functionality, proving it works. Acceptance Criteria emerge during conversations with the Product Owner and Stakeholders and applies to only one Story. Because the Product Owner owns the User Story, the Product Owner has the final say on the Acceptance Criteria.
- That's great for the User Stories, but how do we know we're really done? That's where the Definition of Done (DoD) comes in. DoD is created prior to the first Sprint. It describes the minimum software quality standard and should eventually represent a Releasable Increment standard. DoD applies to all Stories and is continuously improved over time and during the Sprint Retrospective.

What is Definition of Done (DoD)?

A Scrum Team must decide on and document its DoD:

- It is important that everyone have the same understanding of DoD:
  - "Done" is not enough—done is different to different individuals.
  - DoD is what makes a Story ready for Sprint Review.

"DoD" may vary by role

Role	Example of "done"
Coder	DoD may mean code complete.
Stylist	DoD may mean everything looks good and the style sheets have been followed.
Quality Assurance	DoD may mean that all tests have been executed.

- The Scrum Team (Development Team and Product Owner) decide on DoD.
- The Scrum Master facilitates the process but does not vote on DoD.
- It is the Scrum Master's duty to validate with the Development Team upon Story completion that the Story has met DoD before presenting it to the Product Owner.
- The Product Owner will use the DoD to determine if a completed Story is accepted or rejected.
- The Product Owner has the authority to accept or reject a completed Story.

What is Definition of Done (DoD)? (Cont.)

A Scrum Team must decide on and document its DoD:

- The Scrum Team defines DoD.
- The Development Team uses the DoD to hand over the Story to the Product Owner for acceptance.
- Once consensus has been reached, the Scrum Master records the DoD and posts it to be seen by anyone within the organization (full transparency).

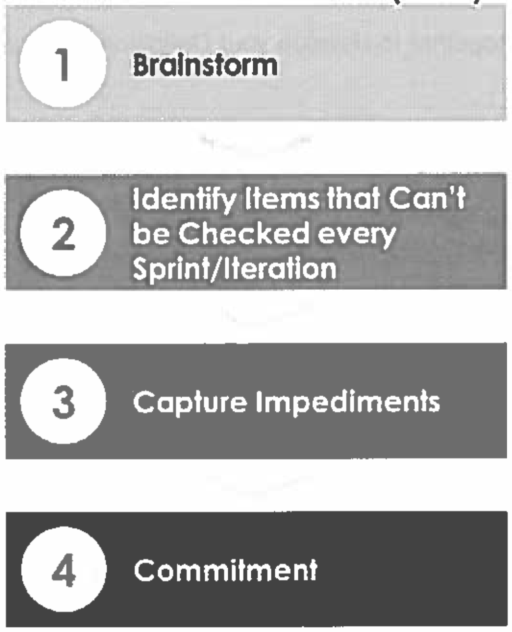
Example:

This is an example of a list of items that would be checked off after determination is made that they are done:

- ✓ Coded
- ✓ Checked in
- ✓ Styled
- ✓ QA'd
- ✓ Product Owner over the shoulder checked
- ✓ Built
- ✓ Promoted

- The Product Owner must participate in the decision of the DoD and gain consensus with the Development Team on what will be accepted and what will be rejected.
- The DoD will vary from Development Team to Development Team.
- The final DoD for a Team should be posted and used to determine acceptance of a completed Story.
- If multiple Teams work on the same product, then all of those Teams should mutually create the DoD.

Suggested Steps to Create Definition of Done (DoD)


1 Brainstorm
2 Identify Items that Can't be Checked every Sprint/Iteration
3 Capture Impediments
4 Commitment

- This is one way to create a Definition of Done. The important thing is to think through what done means and for the Team to come to agreement.
- Suggested steps to create the Definition of Done:
    1. **Brainstorm:** Write down everything essential for delivering on a feature, Iteration/Sprint, and Release; for example, code is complete and checked in, Product Owner signoff, and updated release notes might all be essential pieces of delivering on a feature. Write one item per sticky note. As you brainstorm, think about what it means for that feature to be shippable to ensure that you catch everything that is essential for delivery. Think also about each of the different roles—done for a tester might bring up different items than done for a developer.
    2. **Identify items that can't be checked every Iteration or Sprint:** Look at each of the items on the sticky notes and identify whether or not they can be done at every Iteration/Sprint for each feature being delivered. If they can't be, remove them from your Definition of Done.
    3. **Capture impediments:** For each item that cannot be checked every Iteration or Sprint, discuss the obstacles that keep the Team from delivering this each Iteration or Sprint deliverable. Frequently, these obstacles create issues such as having an unpredictable release period after the last feature is added. Even if the obstacles can't be removed right away, over time, they can be removed and the item can be included in the Team's Definition of Done.
    4. **Commitment:** Get a consensus on the Definition of Done. Go through each item that is completed each Iteration or Sprint.

Class Exercise: D

Overview: In your Team

Instructions:

- Have one persor
rest as develope
- Be prepared to s

Definition of Done
Develop Unit Test
Coded, documented
Peer Reviewed by other Developer
Peer Reviewed by Security Analyst
Functional System Testing
UAT
- affiliate API testing
- developer API testing
UI/UX Review
Product owner review

Test Geo - redundancy
Pass Unit testing
Compiled / Built
Deployed / Promoted
Post deployment review

Assessment: Definition of Done

Questions:

1. True or [✓ False]? The Definition of Done is created by the Product Owner.
2. [✓ True] or False? Definition of Done is what makes a Story ready for Sprint Review.

TEKsystems
Our people make IT possible.

Manage the Product Backlog

Section 13

Review: Product Backlog

- The Product Backlog: single source of requirements for changes to the product. The Product Backlog constantly evolves, adding details that stem from ongoing conversation.
- To prepare for Release Planning, we need a basic Product Backlog. It should include at least a minimal amount of detail in a list of Product Backlog Items ordered by business value.
- Product Backlog management includes communicating items, ordering, optimizing for value, ensuring transparency of the Product Backlog, and ensuring that the Dev Team understands the items to the level needed:
  - The Product Owner may do the above work, or have the Dev Team do it; however, the Product Owner remains accountable and responsible for the completion of the work.

- The Product Backlog is the single source of requirements for changes to the product:
  - Backlog Items are an ordered list of the features, functions, User Stories, requirements, enhancements, and fixes that ***might*** be in the product someday.
- The Product Backlog can live in a spreadsheet, Word document, Agile project management tool, etc.
- There is no prescribed template or formula for the Product Backlog.
- Product Owners are empowered to capture this in whatever medium works best for them and the Development Team.
- As the Product Owner, Development Team, and key stakeholders discuss the Product Backlog Items, this collaboration may change the order of the Items as we uncover dependencies, priority, business value, etc.
- Some may use a document, a spreadsheet, an Agile project management tool, a list on a shared collaboration tool, etc.

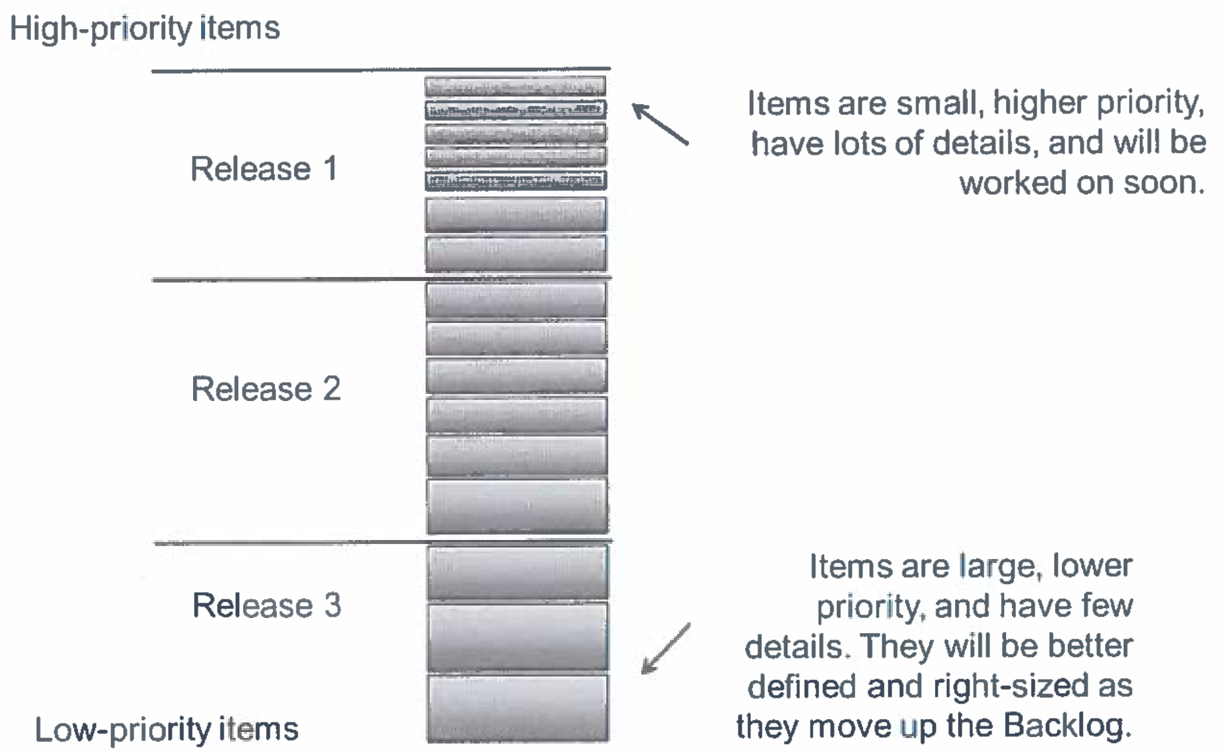
Exercise: Build the Initial Product Backlog

20 min

Instructions:

1. In your Scrum Teams, create a Product Backlog.
2. Product Backlog Items can include User Stories, non-functional items, fixes, technical foundation items, etc.
3. Use sticky notes, index cards, or paper to capture your Product Backlog Items.
4. Create between 10 and 20 Product Backlog Items, splitting them as necessary.
5. Do not worry about ordering or Estimation just yet—we will learn those techniques in subsequent exercises.

Ordering the Product Backlog


High-priority items
Release 1
Release 2
Release 3
Low-priority items
Items are small, higher priority, have lots of details, and will be worked on soon.
Items are large, lower priority, and have few details. They will be better defined and right-sized as they move up the Backlog.

- Jim Johnson, Chairman of Standish Group, reported at the Third International Conference on Extreme Programming (XP2002) that in typical software systems 64% of features are never or rarely used in reality. The most effective way to reduce software cost is to prioritize and deliver only highly used value features.
- Prior to Release Planning, the Product Owner orders the Product Backlog by Business Value. Business Value can be determined by a financial model or a combination of a financial decision and what will bring customers the most satisfaction or value, or drive the most use. Highest priority is given to Stories with the highest business value.
- The Product Owner can use whatever technique for arriving at Business Value that works best for them and the organization.
- The Product Owner can gather input from key stakeholders, SMEs, customer service, customer focus groups, their Scrum Team, surveys, and so on.
- The order may change after collaboration with the Development Team, but the intent is to rank the list from highest value items at the top to lowest value items at the bottom so time is not wasted on low value items during Release Planning.
- This ordered Product Backlog is used to determine what Stories will be included in each Release.