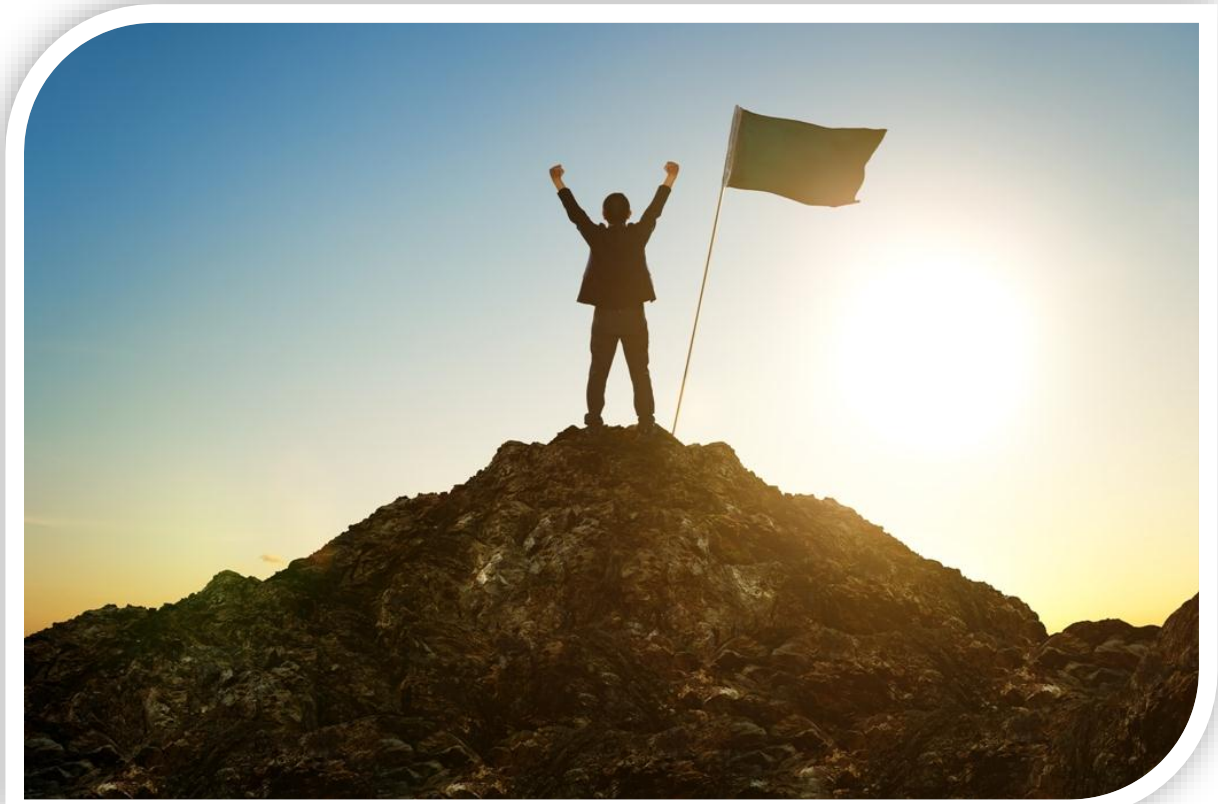




Successful Scrum Adoption



Guidance for Teams – Product Backlog

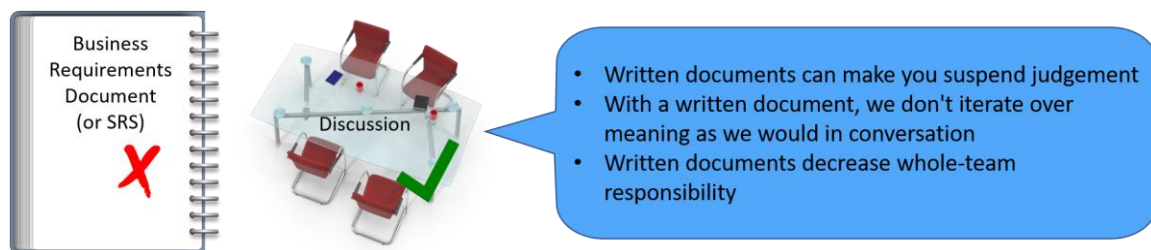
- Product Backlog contains all the requirements of the project.
- Requirements will be in the form of User Stories.
- Let us look at what changes an organization should make to effectively work with Product Backlog.

Shift from Documents to Discussions

- In traditional projects, requirements are exhaustively documented and shared with team members.

Common problems with focus on extensive documentation

- This practice of extensive documentation limits the team members' interaction with Business users. It also creates few other problems.



Written documents can make you suspend judgement

- Written documentation may appear formal, official, and complete.
- By reading the document, team members may blindly follow what is written without even applying their judgement.

With a written document, we don't iterate over meaning as we would in conversation

- Let us consider an example.
- Requirement: "The Parking Lot must have Entry and Exit signs in bold, large attractive font so any one can see them clearly."
- Programmers might interpret attractive font as they like. The meaning of attractive font is taken for granted here.
- Programmers must now ask business users further questions to understand the meaning better: What exactly is an attractive font? What should be the size of letters? In which language should these words should appear?

Written documents decrease whole-team responsibility

- Scrum teams embrace and work on whole-team responsibility.
- Written documents create sequential hand-offs (for example, from Analyst to Programmers, from Programmers to Testers, etc.) and discourage two-way communication.

- This creates a feeling in team members that they are working toward meeting what is written in the document rather than toward success of the project.

No documentation required in Agile?

- Agile manifesto says, “working software over comprehensive documentation.” Does this mean there would be no documentation in agile?
- If you answer “Yes,” that is a ***misinterpretation***.
- Agile development has the goal of finding the right balance between documentation and discussion.
- Agile projects should maintain the required documentation, such as requirements in the form of User Stories, test plans, code, etc.

User Stories

- User Stories encourage the teams to shift the focus from writing about features to talking about them.
- User Story is a short, simple description of a feature told from the perspective of the person or role who wants that new feature.

User Story format:

As a <role> I want <requirement> so that <benefit or reason>

Example: As a supplier, I want to login to the system so that I can see orders placed against my company.

- User Stories can be written on 3" by 5" index cards or can be stored on an electronic backlog management tool.

Progressively refining requirements

- In Scrum projects, we don't collect all requirements upfront.
- While visualizing the product, Product Owner would come up with requirements.
- These requirements can change any time in a Scrum project.
- As the project progresses, requirements will emerge.

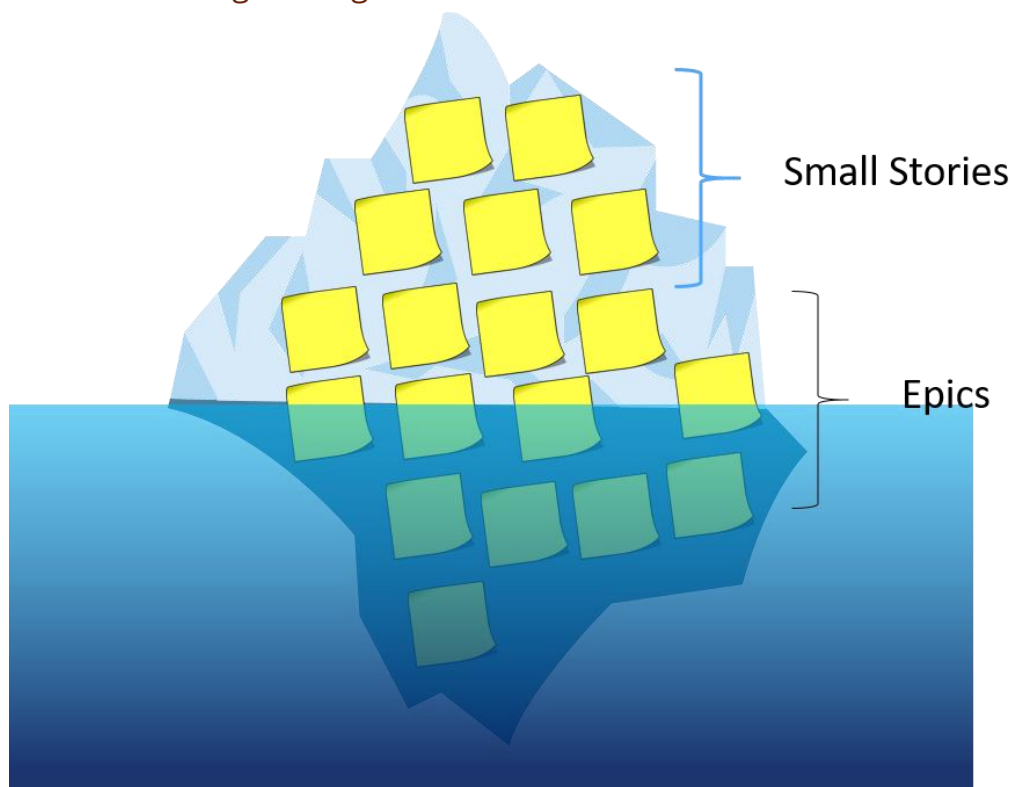
Emergent Requirements


- As the users start working on the system being developed, they may unlock a few other requirements.



- For example, A business user is testing a login page. While attempting to login to a system by providing Username and Password, the business user may find that it would be nice if there is a single sign on feature, that is, if the system automatically logs in the user based on underlying Operating System user (the user account used to log in to the computer or laptop on which this test is executed).
- On traditional projects, emergent requirements are handled in a couple of ways:
 - Adding buffers to schedule so that any emergent requirements can be delivered using those buffer times
 - Establishing a separate Change Budget dedicated to fund these emergent requirements.

Product Backlog Iceberg



Each  represents an User Story

- Product Backlog contains features written with different levels of detail.
- User Stories at the top will be small and have required details so team members can build, test, and integrate them in a single Sprint.
- User Stories at the bottom are large and may have fewer details. These are called Epics.

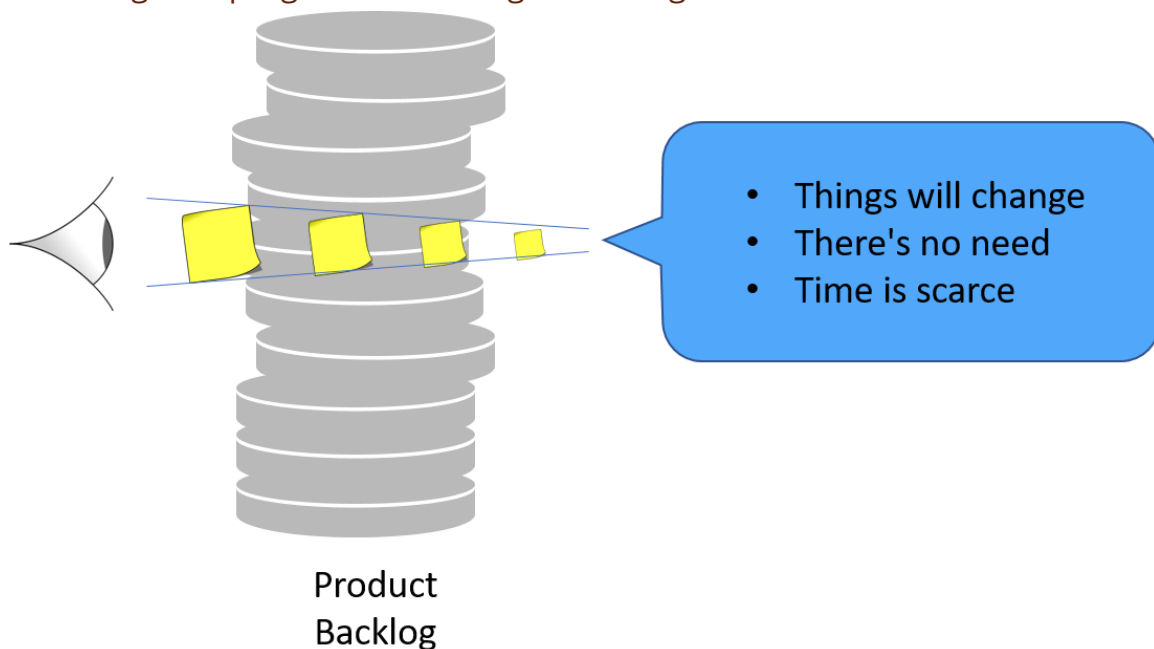
Product Backlog grooming

- Product Backlog items are groomed or looked after as the team works on the Sprint.
- Backlog grooming includes adding new items as they emerge, adding more details to existing items so they can be picked up in upcoming Sprints, removing items that are no longer needed, reprioritizing items, etc.
- Product Owner leads the Backlog grooming exercise.

As a thumb rule, 10% of the effort in each Sprint should be spent in grooming the backlog for upcoming Sprints.

- Conversation about Backlog items can happen at any time among any team members.

Advantages of progressive refining of backlog items



Things will change

- In any project, change is inevitable.
- The iceberg structure of our Product Backlog helps us to keep requirements that can be developed, have full details ready to be worked on.
- Changes are added to Product Backlog as they come, they are prioritized, and details added as part of Backlog grooming in future.

There's no need

- Upcoming items have enough visibility, providing teams with what is required in the **near** future but not the **far** future.
- We can avoid issues by not taking up projects that lack visibility about the **far** future.

Time is scarce

- Nearly all projects have time constraints.
- Treating all requirements as useful and equivalent doesn't help project teams to deliver ideal business value.
- We must make the most out of a limited supply of critical resources.
- When these critical resources are focused on what has more priority for customers in each Sprint (which is timeboxed, that is, constrained by time), teams will be able to deliver value at the end of each Sprint.

Adding Conditions of Satisfaction

- User Stories will have Conditions of Satisfaction—a high-level acceptance test—that must be satisfied for the User Story to be considered complete.
- Conditions of Satisfaction (also called as Definition of Done) are measurable, achievable conditions that will be tested when the User Story is ready for testing.
- When progressively refining User Stories, Conditions of Satisfaction are added when the Business Users think of them.

Learning to start without a specification, that is, requirement

- Scrum teams start the project without exhaustively documented requirements.
- Teams must learn how they can become comfortable with almost no upfront requirements and very short requirements in the form of User Stories. They must learn to talk to Product Owners to discover more details about the User Stories.

Specify by Example

- Instead of writing elaborate and exhaustive requirements documents, teams can use Specification by Example.
- In Specification by Example, requirements are specified by quoting examples.
- Real-world examples help team members to communicate better because they can easily relate to the examples and spot any inconsistencies.
- For example:



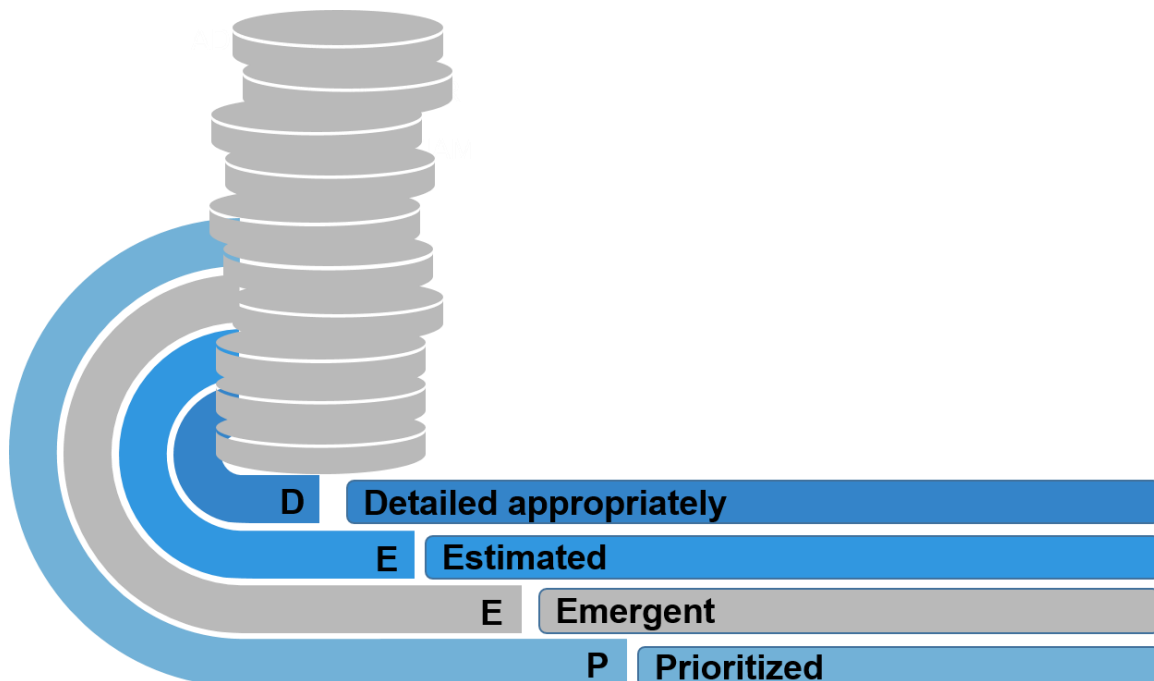
- Requirement: Order values less than \$10,000 can be approved by local managers. Any order with value more than \$10,000 needs two levels of approval – local manager first and then regional manager.
- Product Owner provides an example for this requirement as shown below:

Order Value	Local Manager Approval	Regional Manager Approval
8500	Yes	Not required
10000	Yes	Not required (requirement clearly says greater than 10000)
11780	Yes	Yes

- Specification by Example is very useful to generate automated tests.

DEEP Product Backlog

- This topic was presented in Product Management additional reading material.
- Let us quickly recall here.
 - **D – Detailed appropriately**
 - **E – Estimated**
 - **E – Emergent**
 - **P – Prioritized**



Detailed Appropriately

- Product Backlog Items are detailed appropriately – This means that the items with higher priority are provided with more details when compared to items with lower priority.
- This ensures that the items that might be taken as part of next Sprint are ready with more details than the ones that might be implemented in much later Sprints.
- Throughout the project, requirements are discovered – so the lower priority items will get enough detail as we progress through the project.

Example

Let us observe two Product Backlog Items shown below as an example. The items are shown with an ID for illustration purposes.

1. ID: 76435

As a supplier, I want to access the partner portal so that I can see which orders I must work on and decide for requested material in required quantity

Acceptance criteria:

1. Supplier must already be registered in the system
2. Supplier should be able to see only the material that is allotted to his or her company. No supplier should be able to see material allotted to other suppliers.
3. Procurement Manager should be able to know which supplier has looked at material requests.

This Product Backlog Item is described **with enough details** for the development team to work on in next upcoming Sprint.

2. Id: 80923

As a Procurement Manager, I want to see the list of suppliers <and TBD> and the materials requested against them so that I can plan my activities and <TBD>.

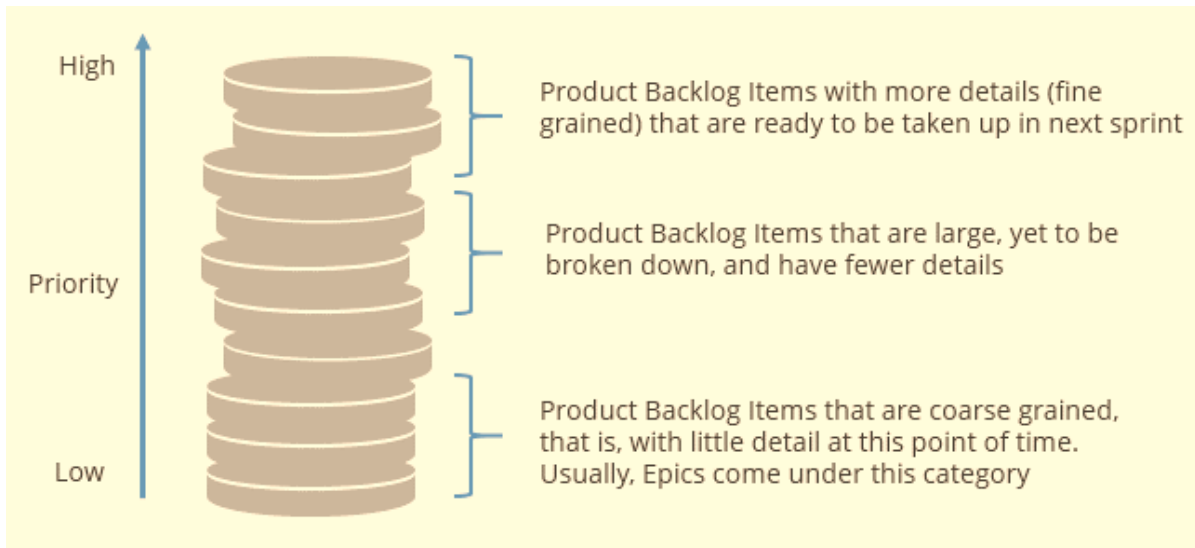
Acceptance Criteria:

1. TBD (TBD indicates “To Be Decided”)

This Product Backlog item is of low priority. It **may not have enough details** at this point of time as it will be implemented in later Sprints.

- This “D” is shown graphically below:





Estimated

- Product Backlog Items are estimated.
- Estimates are expressed in Story Points or Ideal days.
- Task-level details and effort estimation for these items will be done as part of Sprint Planning and Sprint Backlog (which is a subset or child of Product Backlog) will contain more details and tasks.

Emergent

- Product Backlog is **live** throughout the project. This means that the Backlog items can be added, moved, modified, or removed as required by the customer.
- The backlog changes continuously throughout the project.

Prioritized

- All Product Backlog Items **must** be prioritized.
- High-priority items will be on top and will be implemented first. Lower priority items will be at the bottom.
- Once a Product Backlog Item is completed, it will be removed from Product Backlog.