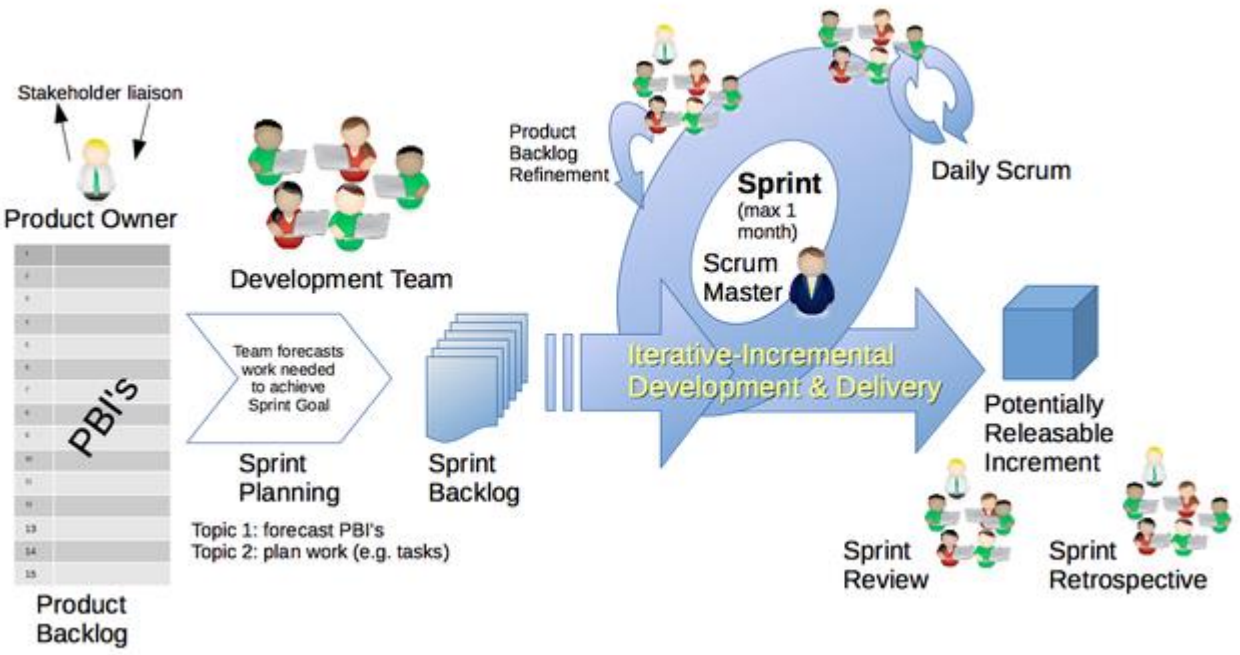
Backlog Refinement and Estimation



SWEN-261 Introduction to Software Engineering

Department of Software Engineering
Rochester Institute of Technology

Before a story can be placed in the sprint backlog, it must be refined and given an effort estimate.

- The user story statement alone provides no detail about what is required or how to implement it.
- Backlog Refinement (also Backlog Grooming) adds those details.
  - ***Acceptance criteria provide details of requirements***
  - ***Solution tasks outline how to implement the story to satisfy the acceptance criteria***
- Based on the solution tasks, the developers can estimate story points using Planning Poker.
- With enough stories having story points, sprint planning can fill the sprint backlog up to the team's velocity.

Acceptance criteria come from the Product Owner or user representatives.

- Defining acceptance criteria can be done in brainstorming sessions with the product owner
  - ***The Product Owner leads the discussion and drives the exploration of the acceptance criteria.***
  - ***The developers ask questions to further elaborate the acceptance criteria.***
  - ***This can be done all together or in smaller groups discussing subsets of user stories.***

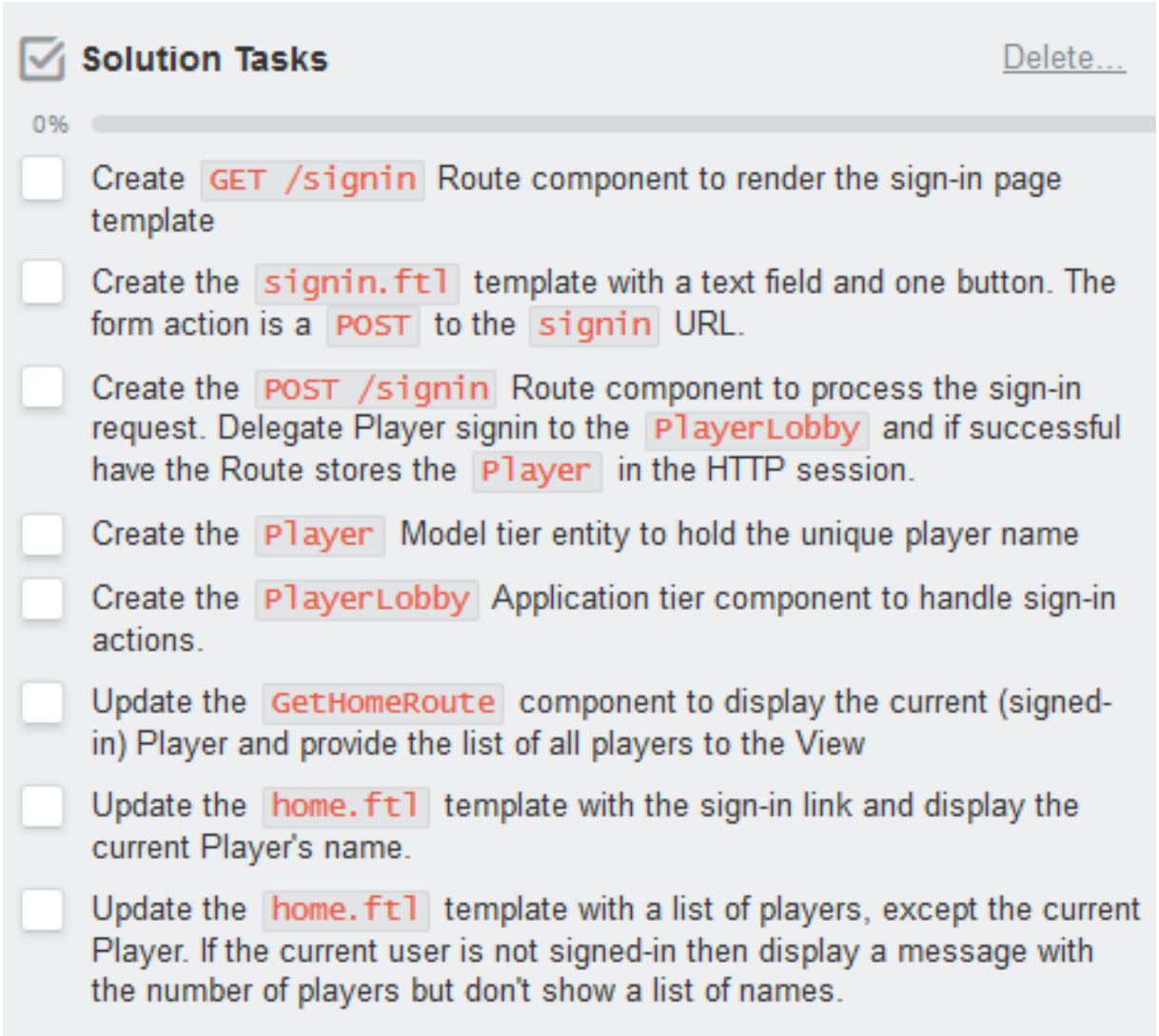
Let's review:
What makes good acceptance criteria?

- Like stories, focus on the *what* not the *how*.
- Use a Given/When/Then format:
  - ***GIVEN some precondition WHEN I do some action THEN I expect some result***
  - *Given* that I'm not signed-in *when* I visit the Home page *then* I expect to clearly see how to Sign-in.
  - *Given* that some other player has already signed-in with my name *when* I attempt to sign-in with my name *then* the system should reject the request with an error message and re-render the Sign-in form.

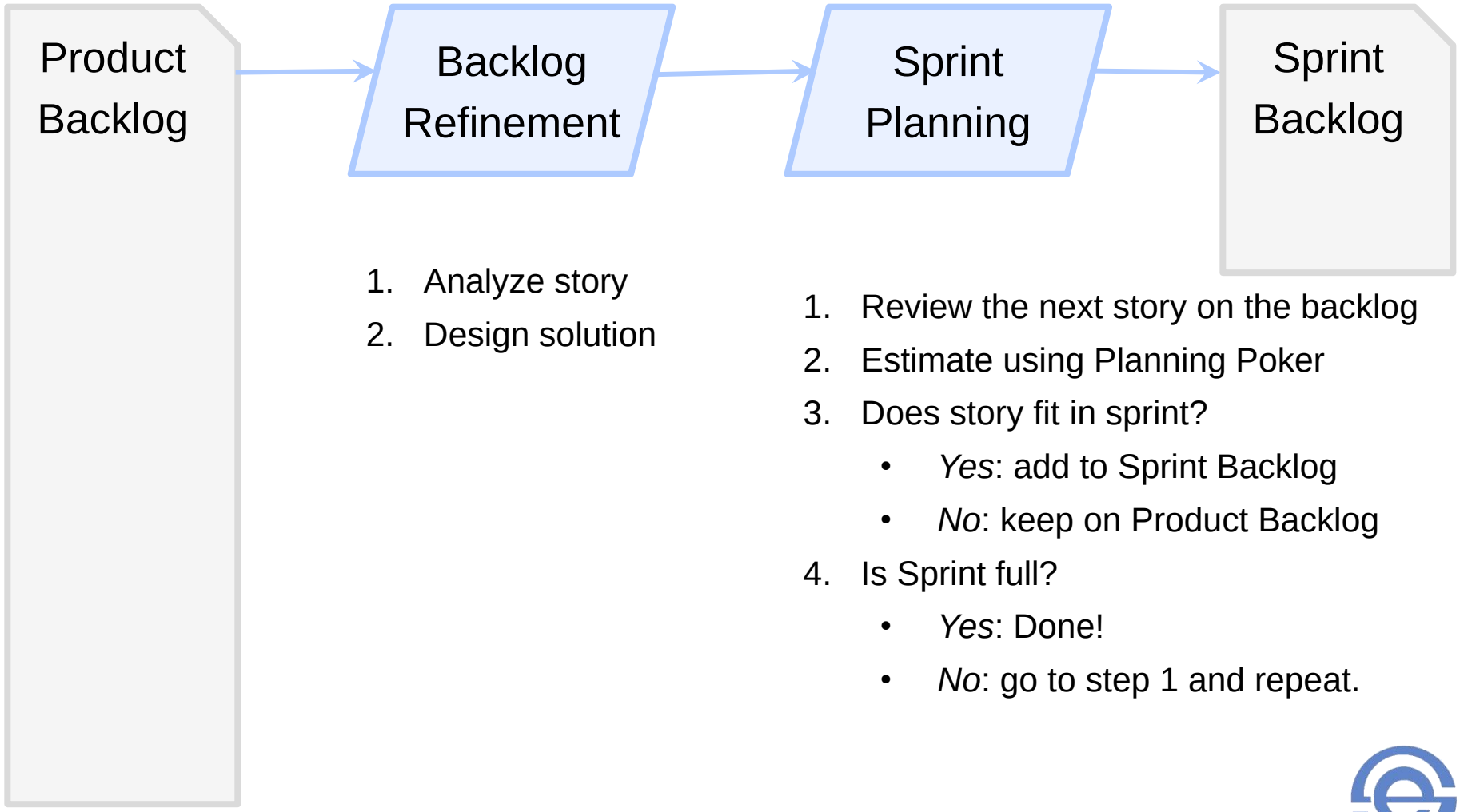
With acceptance criteria defined, a developer then fleshes out a skeleton design.

- Evolve the analysis models:
  - ***Explore new domain concepts***
  - ***Alter existing domain model***
  - ***Does the story alter the web interface? Then update the Statechart with your proposed states.***
- The design is very high-level:
  - ***Create or modify Views and Controllers in the UI tier***
  - ***Create or modify Services in the Application tier***
  - ***Create or modify Entity or Value Objects in the Model tier***
  - ***Create or modify any other helper component***
  - ***Refactoring existing code to improve the overall design***

These descriptions become tasks in the story's Trello card.



During Sprint X you refine stories in preparation for the Sprint Planning meeting for Sprint X+1.



1. Analyze story
2. Design solution

1. Review the next story on the backlog
2. Estimate using Planning Poker
3. Does story fit in sprint?
   - *Yes*: add to Sprint Backlog
   - *No*: keep on Product Backlog
4. Is Sprint full?
   - *Yes*: Done!
   - *No*: go to step 1 and repeat.

Planning poker is a technique devised by Mike Cohn.

- It is a form of *expert estimation* in which every team member is an expert.
- The *points* assigned are abstract; they do not relate to hours of effort.
  - ***A sprint's capacity is not in hours but "level of effort"***
- The point system provides relative levels of effort.
  - ***Small effort: 0, ½, 1, 2 or 3***
  - ***Medium effort: 5, 8 or 13***
  - ***Large effort: 20, 40 or 100***
  - ***Unknown: ?***

OK, but how do you estimate a story, really?

- Create an estimate for each *Solution Task* in your story design.
  - ***Consider the type of component to build (or modify)***
  - ***Consider the complexity of the feature***
  - ***Consider how well you know the technology***
- Add up each task estimate and *round up* to the nearest Poker (Fibonacci) number.
- Expert developers do this calculation implicitly based their large experience base.

Here is an example matrix of component estimation.

Architectural Tier	Component Type	Small / Low	Medium	Large / High
UI	UI View (View Template)	1	3	5
UI	UI Controller (Spark Route)	1	2	3
Application	Service	2	3	5
Model	Entity	1	2	3
Model	Value Object	1	2	3

- Each team member can **independently** estimate a user story by:
  - ***For each class that will get touched/created when implementing the user story, identify its component type.***
  - ***For each class, find its estimate in the chart based on your estimated level of development effort needed.***
  - ***Add up all the class estimates to get your estimate for the user story.***

Here's how Planning poker works.

1. The Product Owner reads the top story on the Product Backlog.
2. The team reviews the acceptance criteria and the suggested solution design.
3. To vote, each player picks the point card for his or her estimate.
4. Players reveal their cards all at once.
5. If there is consensus on one number, you're done.
6. Otherwise:
    1. ***Have the outliers (high/low) explain their position***
    2. ***Team discusses***
    3. ***Vote again until consensus is reached***

What should the team do if no consensus is found?

- There are usually two issues that prevent consensus.
- Product uncertainty:
  - ***The requirements (acceptance criteria) are too vague***
  - ***Send the story back for further (analysis) refinement***
- Technical uncertainty:
  - ***Identify the uncertainty in the solution design***
  - ***Create a spike story for this sprint to establish certainty***
  - ***Send the story back for further (design) refinement***
- In either situation, the story should stay on the Product Backlog until the uncertainty is resolved.