SIT725 Software Engineering

Week 4

Analysis Modeling for WebApps

Reminder! Analysis Modelling

- The purpose of this Analysis Modelling is to:
  - Build upon information derived during the communication activity
  - Refining and modelling WebApp requirements
  - Ultimately establish a deeper understanding of the nature of our WebApp

- *Discover detail requirement*
- *Document and represent detail requirement*
- *Prepare product backlog*

Requirements Analysis Goals

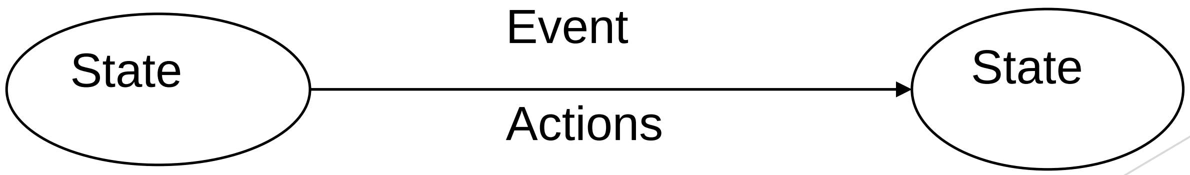
- Specifies the software's operational characteristics
- Indicates the WebApp’s interaction with other system elements
- Establishes constraints that the WebApp must meet
- Provides the WebApp designer with a representation of information, function, and behavior
  - Later divided into different categories:
    - Architectural
    - Interface
    - Class/data
    - Component-level designs
- Lastly, provides the developer and customer with the means to assess quality once the WebApp is built

The Interaction Model

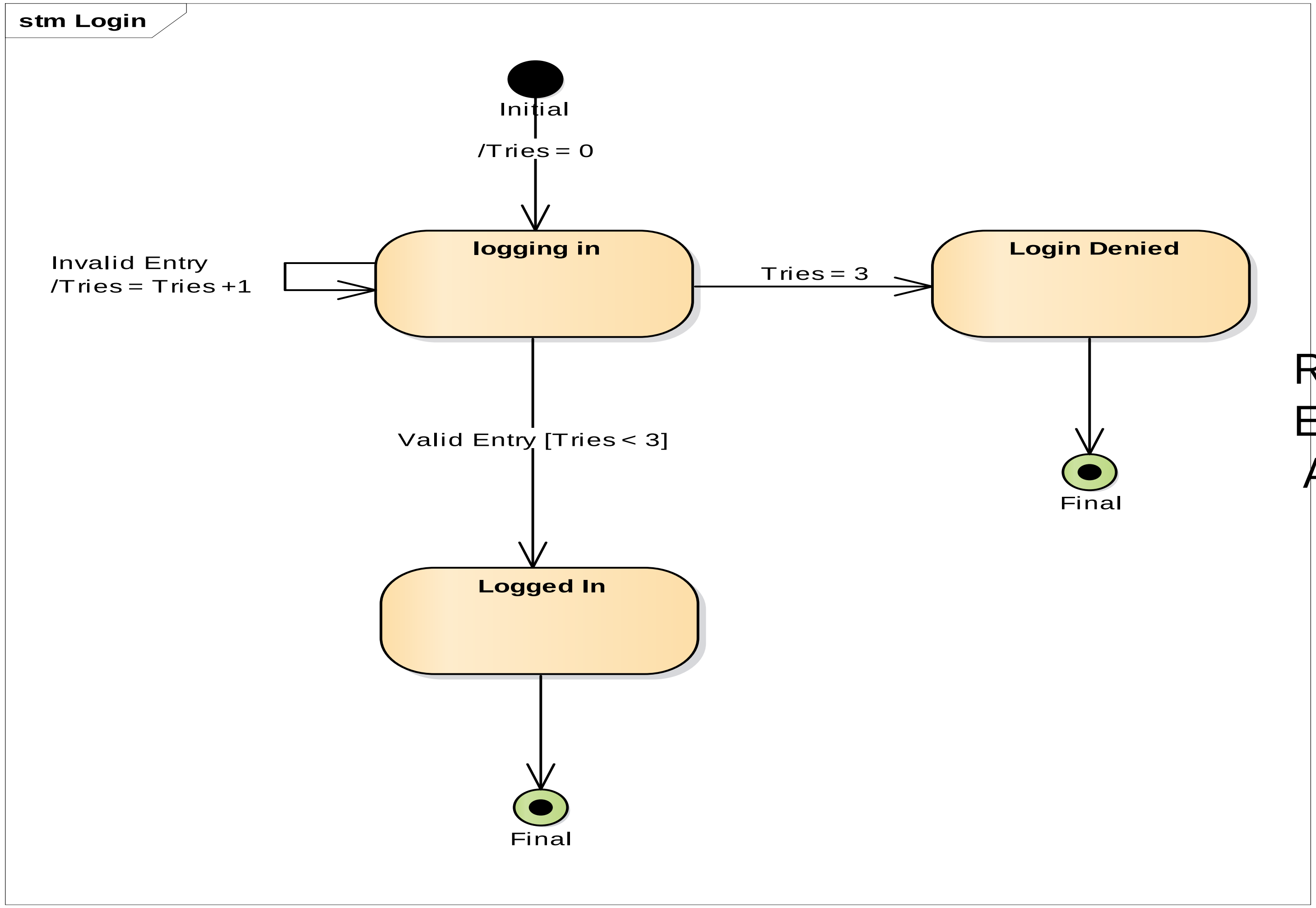
- The interaction model describes how users interact with our WebApp.
- Composed of four elements:
  1. Use-cases
  2. State diagrams
  3. Activity Diagram
  4. Use case description
  - Object orientated modeling

Review of State Diagrams

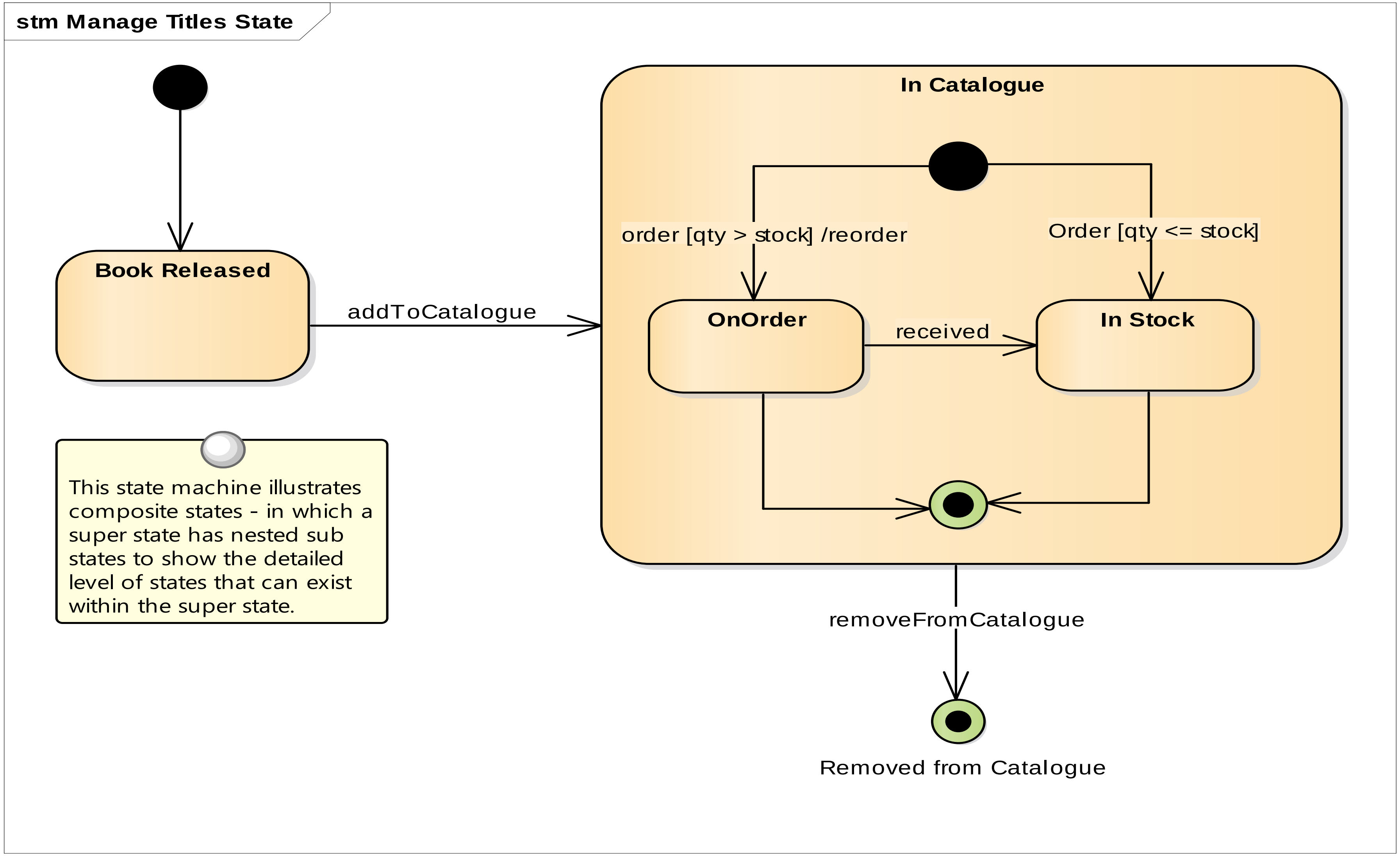
- <u>Three main aspects:</u>
- The possible state of each class
- Events that cause a transition from state to another
- Actions that result from a state change
- Initial state, final state and transition
- Can represent the whole history of a class from its creation to its destruction.



Sate diagram for user log in



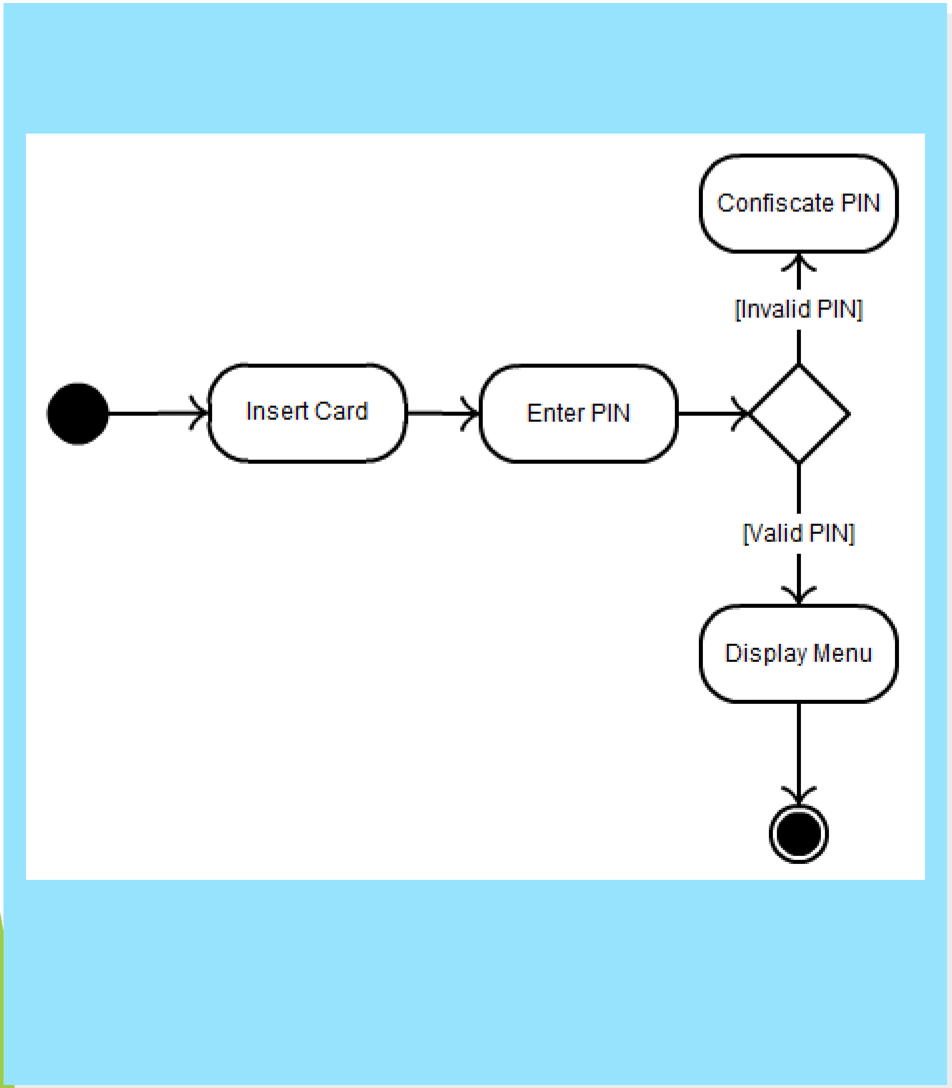
Reference: Enterprise Architect



This state machine illustrates composite states - in which a super state has nested sub states to show the detailed level of states that can exist within the super state.

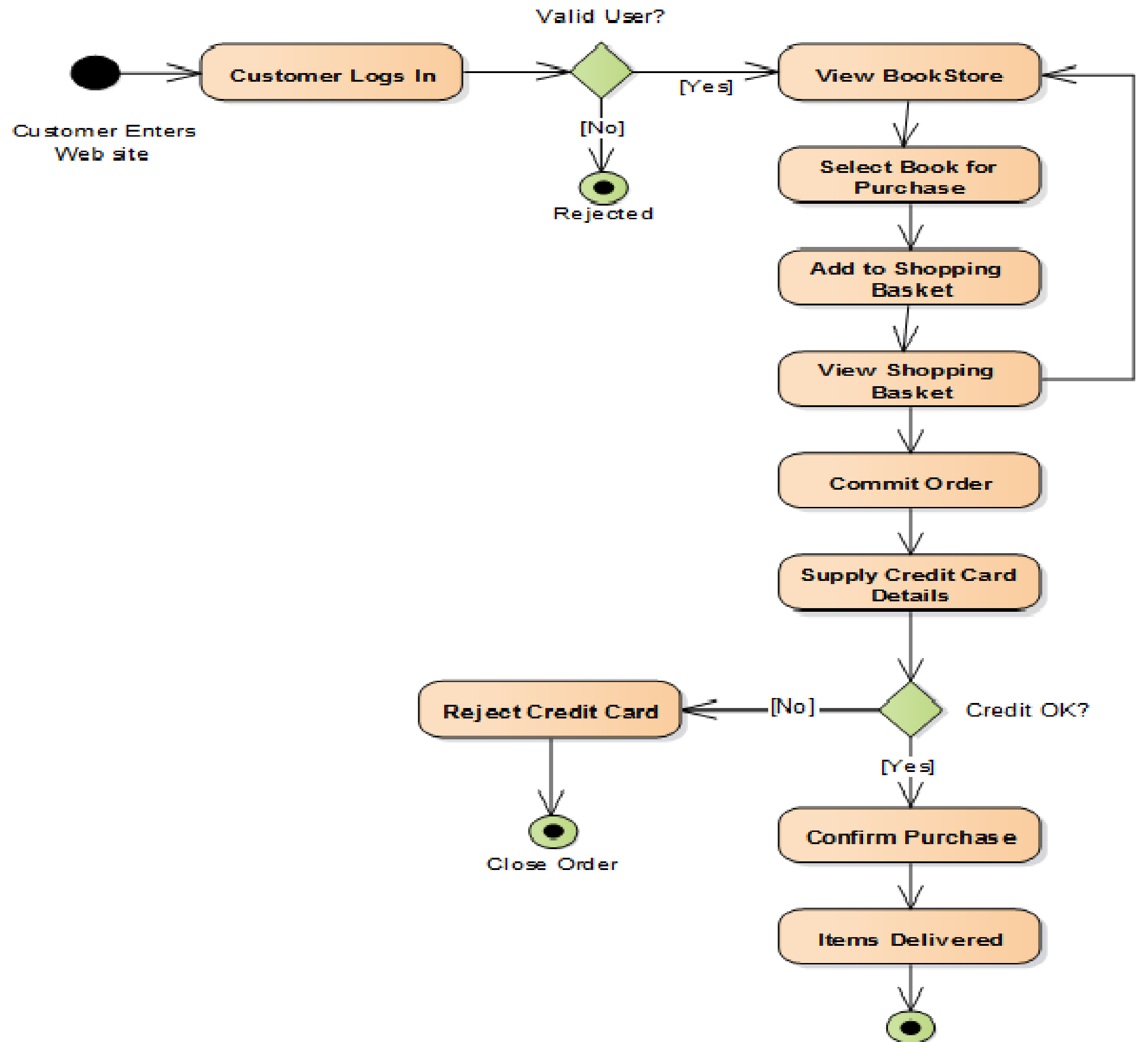
Reference: Enterprise Architect

Simple Activity Diagram



- Rounded rectangle - represent a specific system function/action
- Arrow - represents the flow of control from one function/action to another
- Diamond - represents a branching decision

Simple Activity Diagram



Activity diagram of a simple book order

- Rounded rectangle - represent a specific system function/action
- Arrow - represents the flow of control from one function/action to another
- Diamond - represents a branching decision

The Functional/ Process Model

- WebApp's can deliver a range of computational and manipulative functions:
  - Such as using or producing content!
- In understanding the requirements of the WebApp, the functions required by the user need to be identified!
- All these functions we identify, essentially process information in some way in the system
  - Usually input → process → output
- The functional model is a representation of how information is transformed

The Functional Model

- The functional model addresses 2 processing elements of the WebApp
  - User observable functionality: Any processing functions that are initiated directly by the user
  - What is the processing to be performed by classes
    - Manipulating attributes
    - Collaboration with other classes to achieve a goal
  -
- An activity diagram can be used to represent the processing flow where you include the process classes.

The Configuration Model

- WebApp’s have to be designed and implemented to handle a variety of different environments.
  - What kind of network connection?
  - What server hardware and operating system?
  - Does the WebApp need to access a large database?
  - What browsers will users have?
- Often the configuration model is nothing more than a list of server-side and client-side attributes.
- But… in more complex WebApp’s we should consider more complex aspects like:
  - Loading, caching, remote databases and multiple servers.

Relationship-Navigation Analysis

- With regards to WebApp’s:
    - Each architectural element can be linked to all other architectural elements.
    - Or what mechanisms do we have and what navigation options do they provide?
- This linking of mechanisms of our WebApp and the navigation options they provide is great!
    - However when the volume of content and user interaction complexity increases, so does the navigational complexity.

- *How can we establish the links between content objects and the functions that provide user-required capabilities???*

The Answer? RNA!

- Relationship-navigation analysis (RNA) identifies relationships among the elements uncovered as part of the creation of the analysis model
- RNA approach is categorized into 5 Steps:
  - *Stakeholder analysis*—identifies the various user categories and establishes an appropriate stakeholder hierarchy
  - *Element analysis*—identifies the content objects and functional elements that are of interest to end users
  - *Relationship analysis*—describes the relationships that exist among the WebApp elements (content objects or functions)
  - *Navigation analysis*—examines how users might access individual elements or groups of elements
  - *Evaluation analysis*—considers realistic issues (e.g., cost/benefit) associated with implementing the relationships defined earlier

How Do We Establish Relationships Between Content Objects and Functionality?

- We can use a list of questions to help us assess the relationships among elements (content objects or functions)
  - Which have been identified within the analysis model.

Lets take a look at the questions!

Relationship Questions

1. Is the element a member of a broader category of element?
2. What attributes or parameters have been identified for the element?
3. Does descriptive information about the element already exist? If so, where is it?
4. Does the element appear in different locations within the WebApp? If so, where?
5. Is the element composed of other smaller elements? If so, what are they?
6. Is the element a member of a larger collection of elements? If so, what is it and what is its structure?
7. Is the element described by an analysis class?
8. Are other elements similar to the element being considered? If so, is it possible that they could be combined into one element?

Relationship Questions.. Continued

9. Is the element used in a specific ordering of other elements? Does its appearance depend on other elements?
10. Does another element always follow the appearance of the element being considered?
11. What pre- and post-conditions must be met for the element to be used?
12. Do specific user categories use the element? Do different user categories use the element differently? If so, how?
13. Can the element be associated with a specific formulations goal or objective? With a specific WebApp requirement?
14. Does this element always appear at the same time as other elements appear? If so, what are they?
15. Does this element always appear in the same place (e.g., same location of the screen or page) as other elements? If so, what are they?

How Do We Analyse Navigational Requirements ?

- So we have developed the relationships, now what?!
- We need to consider the requirements that dictate how:
  - Each user category will navigate from one element (e.g. content object) to another.
- We need to consider the overall navigation requirements.
- Again we can use a number of questions to ascertain this information!

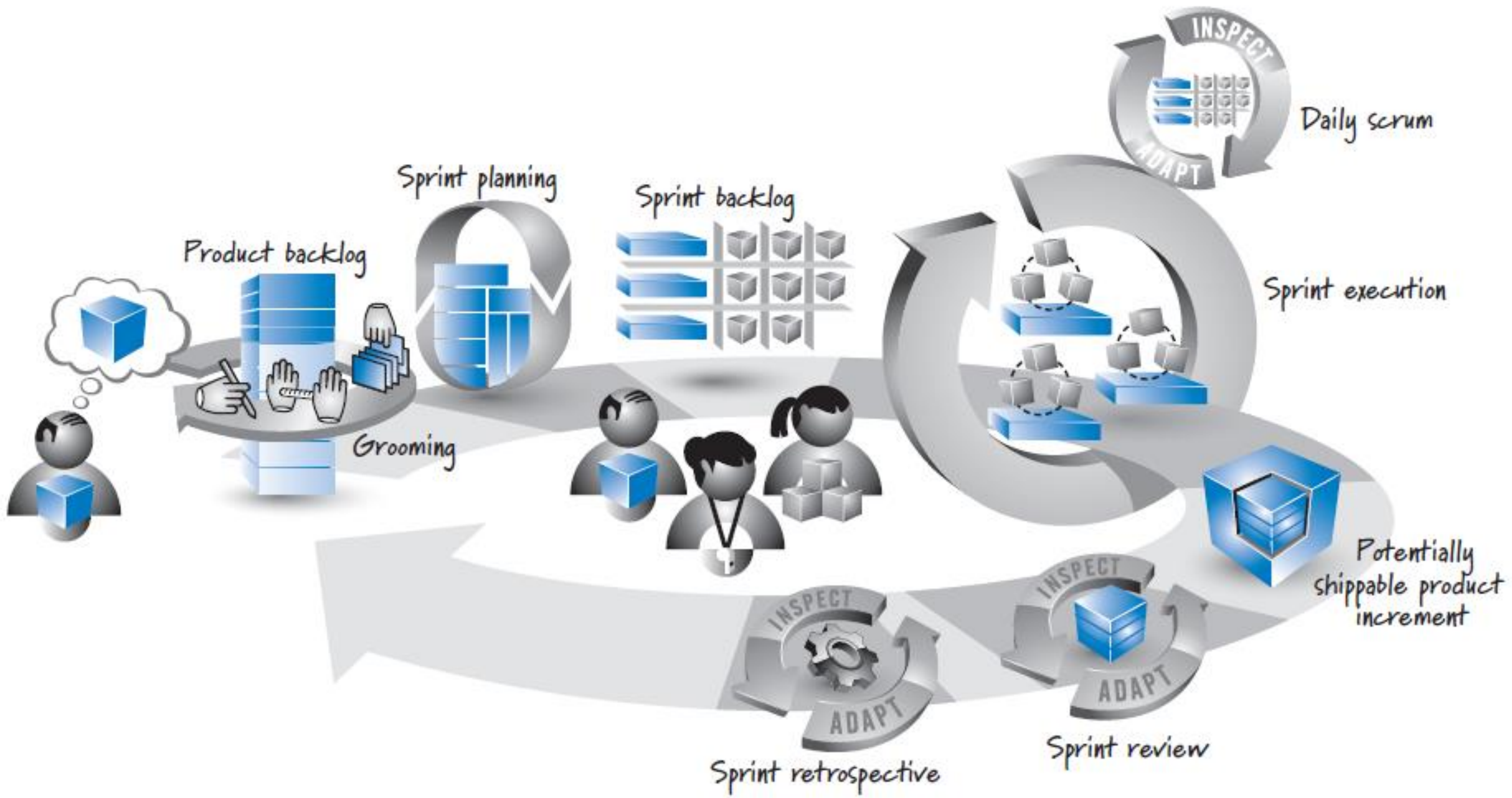
Navigational Requirements Questionnaire

1. Should certain elements be easier to reach (require fewer navigation steps) than others? What is the priority for presentation?
2. Should certain elements be emphasized to force users to navigate in their direction?
3. How should navigation errors be handled?
4. Should navigation to related groups of elements be given priority over navigation to a specific element.
5. Should navigation be accomplished via links, via search-based access, or by some other means?
6. Should certain elements be presented to users based on the context of previous navigation actions?
7. Should a navigation log be maintained for users?

Navigational Requirements Questionnaire

8. Should a full navigation map or menu (as opposed to a single “back” link or directed pointer) be available at every point in a user’s interaction?
9. Should navigation design be driven by the most commonly expected user behaviors or by the perceived importance of the defined WebApp elements?
10. Can a user “store” his previous navigation through the WebApp to expedite future usage?
11. For which user category should optimal navigation be designed?
12. How should links external to the WebApp be handled? overlaying the existing browser window? as a new browser window? as a separate frame?

Detail requirements and documenting requirements



Reference of image: Book-- **Essential Scrum—Kenneth S. Rubin**

Deriving detail requirements from Initial product backlog: Backlog Grooming

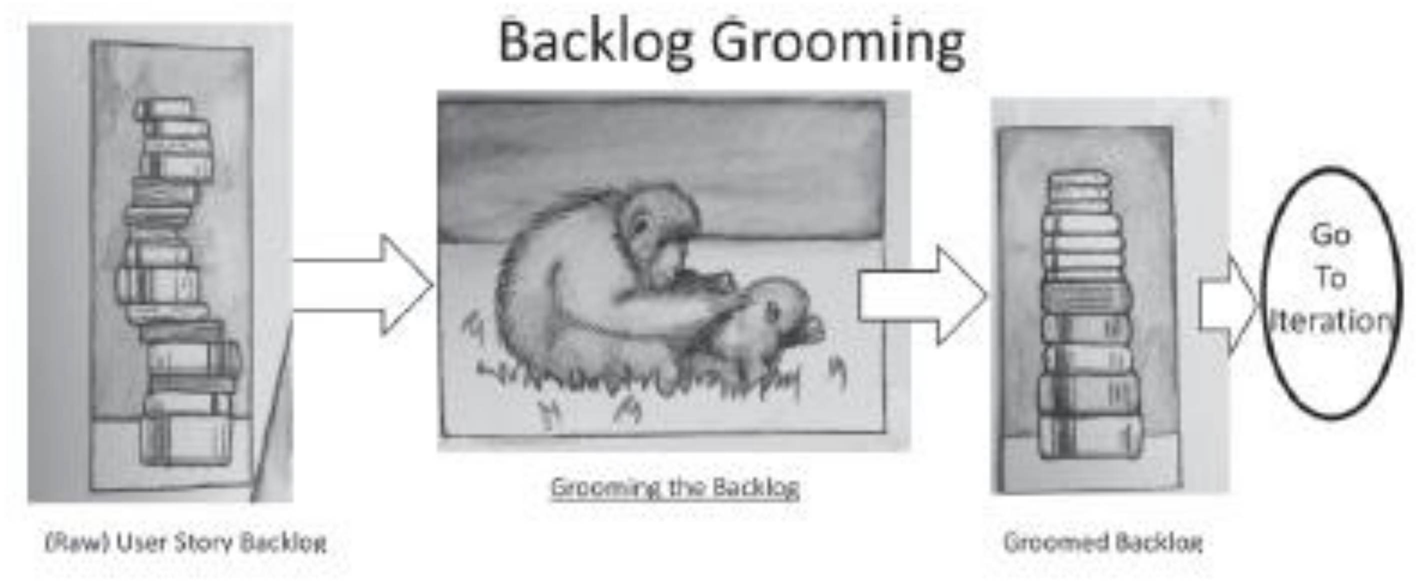
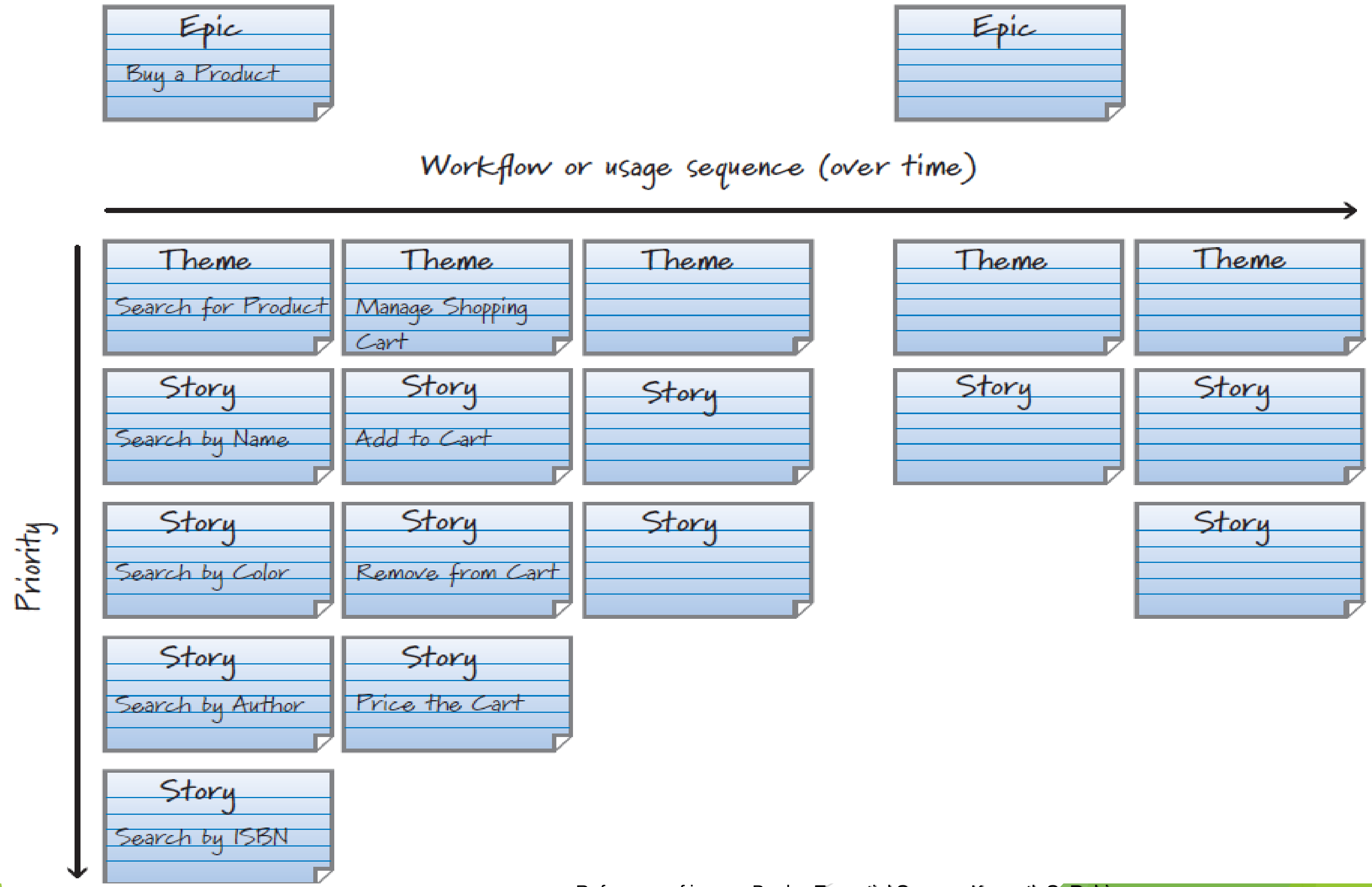


Image---The agile age-Managing projects effectively using agile scrum---Brian vanderjack

- Happens before sprint planning meeting
- Participants: Owner, development team, stakeholders, Scrum master, Scrum team members
- Use prototype, CASE tool, draw use case diagram, use case description, activity diagram
  - Identify detail requirements, prepare the product backlog
  - Estimates the user story, priority is made at this time

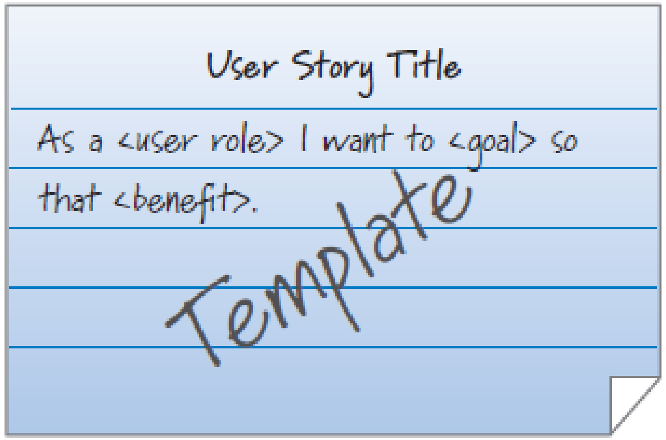
Deriving detail requirements (Story maps)

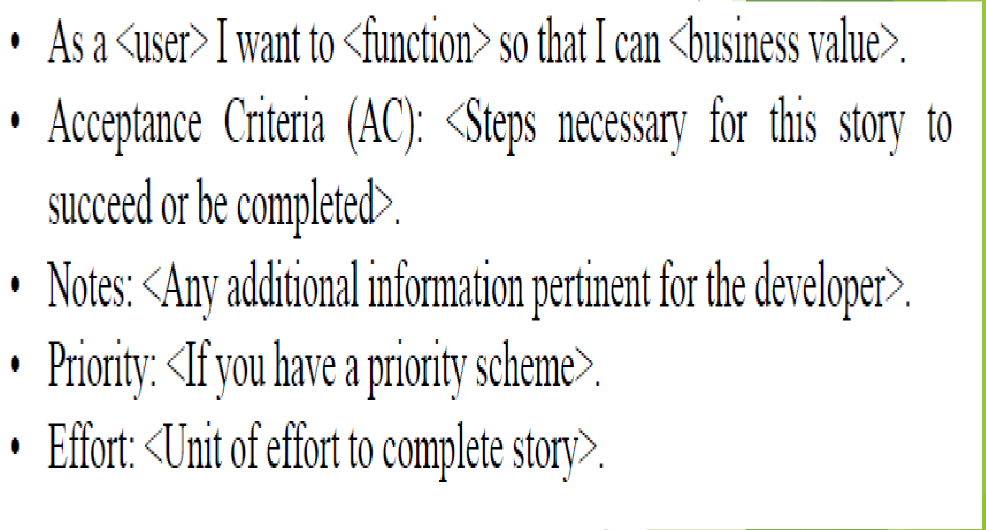


Reference of image: Book-- **Essential Scrum—Kenneth S. Rubin**

User stories

- Your Epics will be decomposed in detail stories according to story map and the models you presented
- **Acceptance criteria: define the boundaries of a user story, and are used to confirm when a story is completed and working as intended.**
-



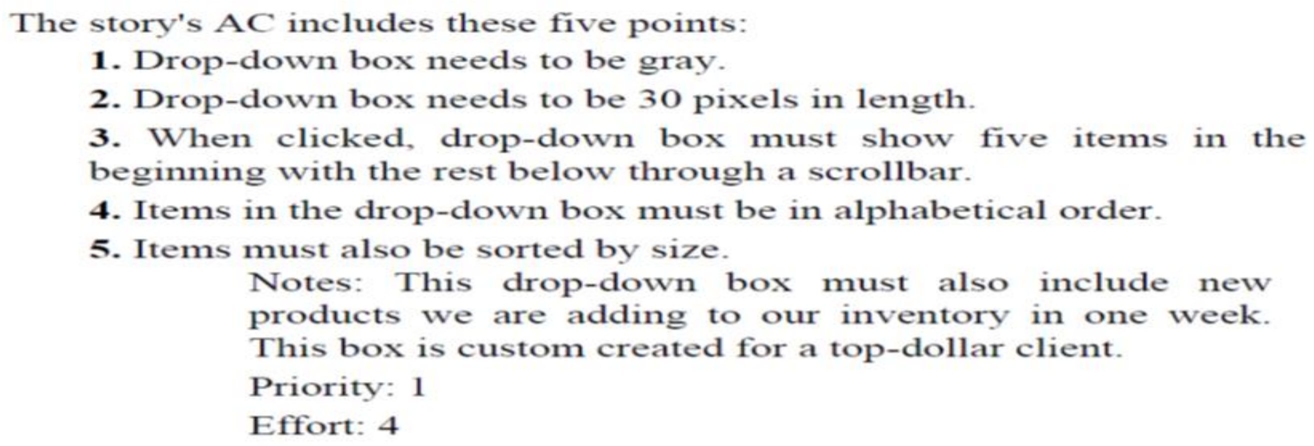


- As a <user> I want to <function> so that I can <business value>.
- Acceptance Criteria (AC): <Steps necessary for this story to succeed or be completed>.
- Notes: <Any additional information pertinent for the developer>.
- Priority: <If you have a priority scheme>.
- Effort: <Unit of effort to complete story>.

Reference of image: Book-- **Essential Scrum—Kenneth S. Rubin**

User stories

- User story-: As a Customer I want to have a drop-down box in my catalogue so that I can easily see a selection of inventory that I can buy
- Acceptance criteria (AC) and other information for- User story-1 :



The story's AC includes these five points:

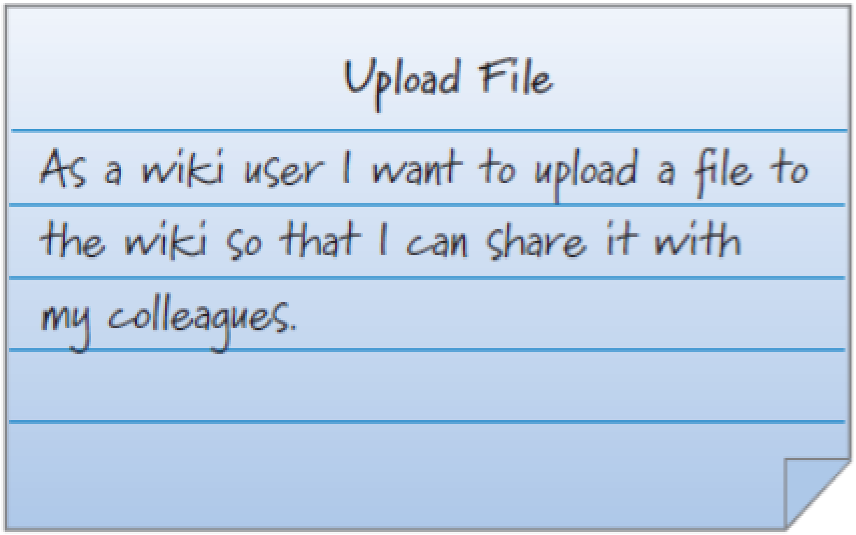
1. Drop-down box needs to be gray.
2. Drop-down box needs to be 30 pixels in length.
3. When clicked, drop-down box must show five items in the beginning with the rest below through a scrollbar.
4. Items in the drop-down box must be in alphabetical order.
5. Items must also be sorted by size.

Notes: This drop-down box must also include new products we are adding to our inventory in one week. This box is custom created for a top-dollar client.

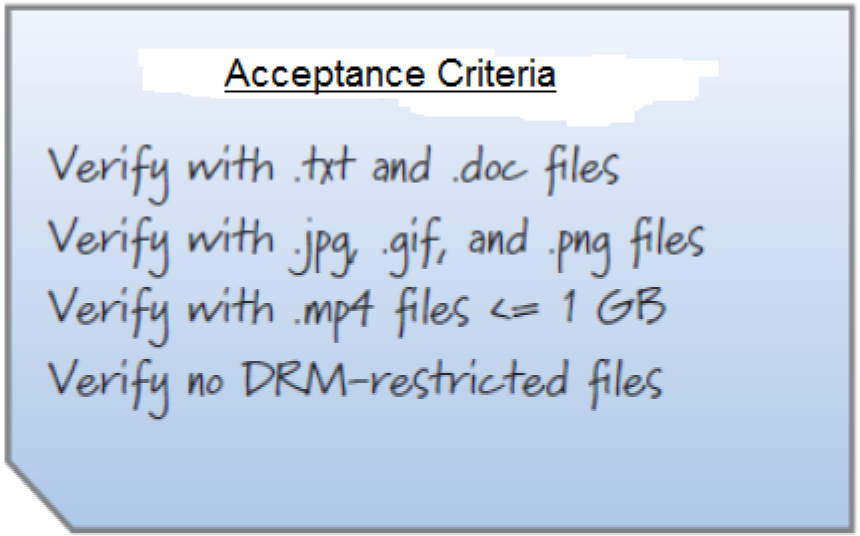
Priority: 1

Effort: 4

More example on user stories


Upload File

As a wiki user I want to upload a file to the wiki so that I can share it with my colleagues.


Acceptance Criteria

Verify with .txt and .doc files
Verify with .jpg, .gif, and .png files
Verify with .mp4 files <= 1 GB
Verify no DRM-restricted files

Reference of image: Book-- **Essential Scrum—Kenneth S. Rubin**

Stories should be Estimatable
Stories should be *estimatable* by the team that will design, build, and test them.
Estimates provide an indication of the size and therefore the effort and cost of the stories
Bigger stories require more effort and therefore cost more money to develop than smaller stories.

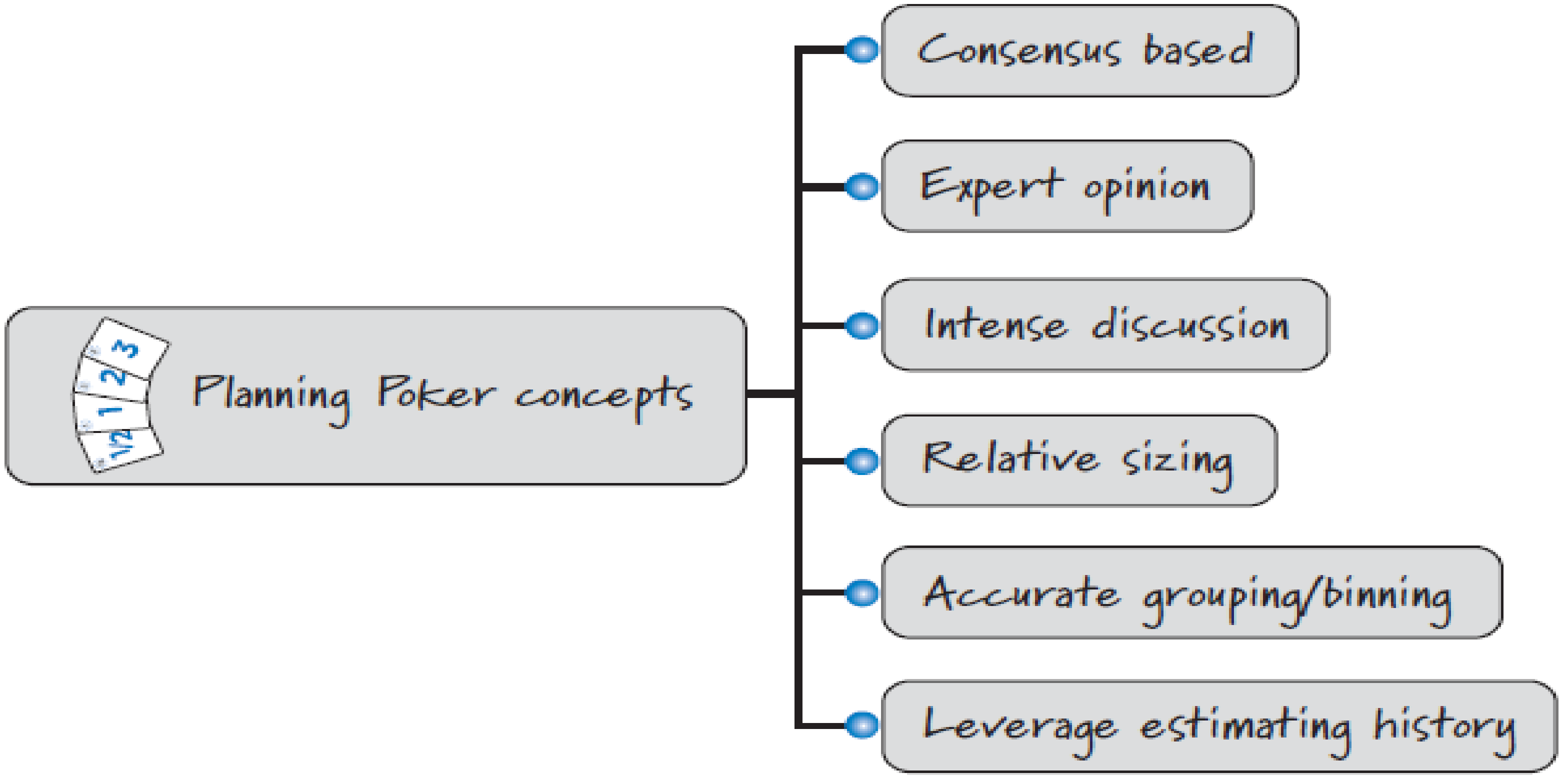
Estimation

- Although there is no standard unit for PBI size estimates, by far the two most common units are story points and ideal days.
- Story points measure the bigness or magnitude of a PBI.
- It is expected story points to be influenced by several factors, such as complexity and physical size.
- Something doesn’t have to be physically large to be big. The story might represent the development of a complex business algorithm. The end result won’t be very large,
  - but the effort required to develop it might be large.
- On the other hand, a story might be physically quite big but not complex.
- Let’s say we have to update every cell in a 60,000-cell spreadsheet. None of the individual updates is difficult, but the updates can’t be automated.

Estimation

- **Ideal Days**
- An alternative approach for estimation is to use ideal days.
- Ideal days are a familiar unit—they represent the number of effort-days or person-days needed to complete a story.
- Planning Poker Technique for estimation
- **Planning Poker** is a technique for sizing PBIs that was first described by James Grenning (Grenning 2002) and then popularized by Mike Cohn (Cohn 2006). Planning

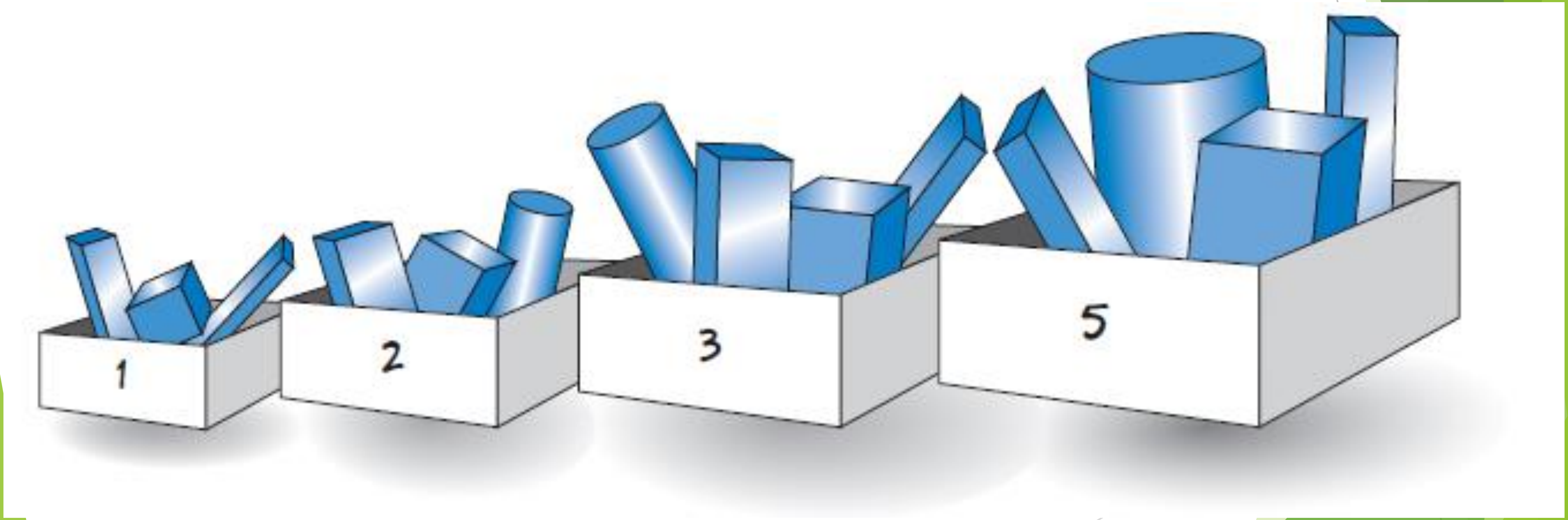
Reference of image: Book-- **Essential Scrum—Kenneth S. Rubin**



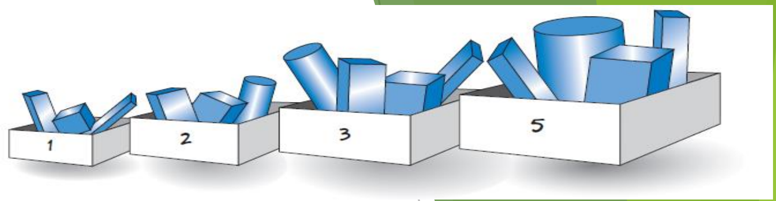
Estimation

- **Estimation Scale and Bining**
- To perform Planning Poker, the team must decide which scale or sequence of numbers it will use for assigning estimates.
- The most frequently used scale is the one proposed by Mike Cohn, based in part on a modified Fibonacci sequence: 1, 2, 3, 5, 8, 13, 20, 40, and 100.
- When using this type of scale, we group or bin together like-size user stories and assign them the same number on the scale.

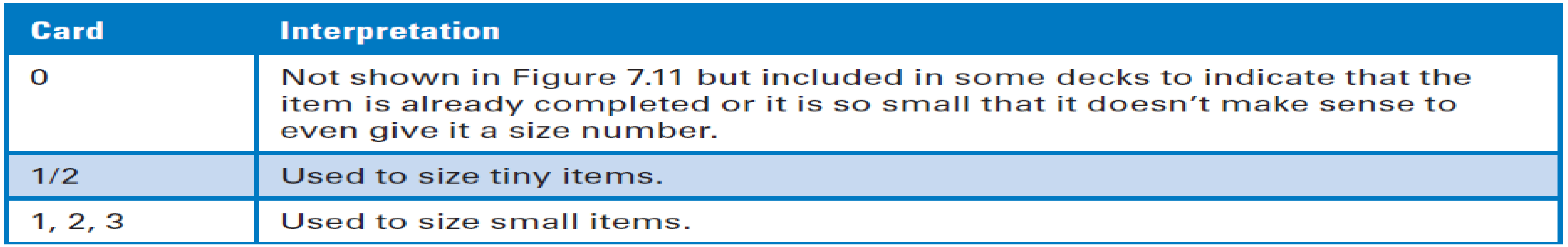
Reference of image: Book-- Essential Scrum—Kenneth S. Rubin



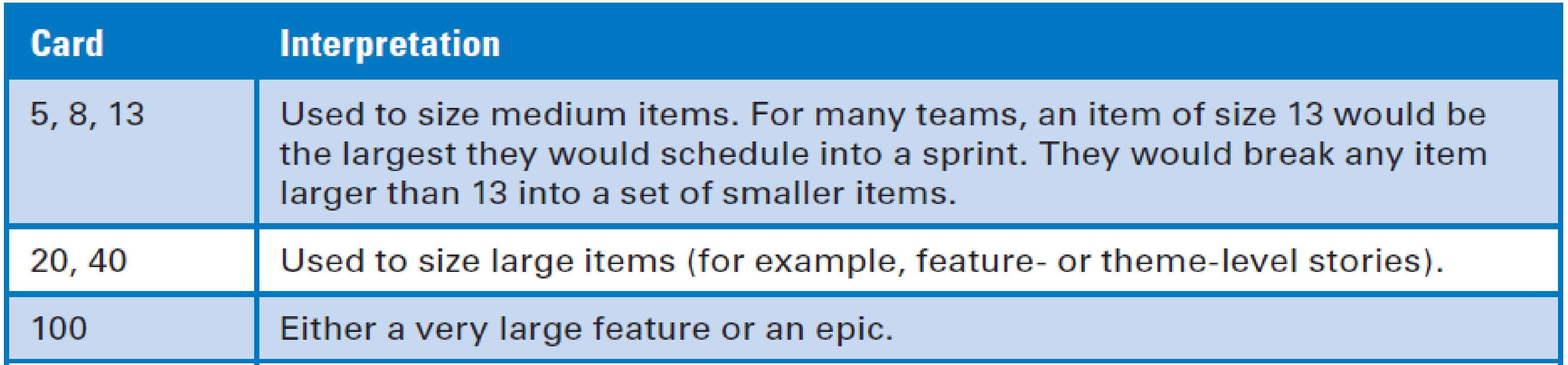
Estimation



- **How to play**
- The full Scrum team participates when performing Planning Poker.
- During the session, the product owner presents, describes, and clarifies user stories.
- The Scrum Master coaches the team to help it better apply Planning Poker.
- The Scrum Master is also constantly looking for people who, by their body language or by their silence, seem to disagree and helping them engage.
- And the development team is collaboratively generating the estimates.
- Each development
- A common interpretation of these cards is described in Table team member is provided with a set of Planning Poker cards



Card	Interpretation
0	Not shown in Figure 7.11 but included in some decks to indicate that the item is already completed or it is so small that it doesn't make sense to even give it a size number.
1/2	Used to size tiny items.
1, 2, 3	Used to size small items.



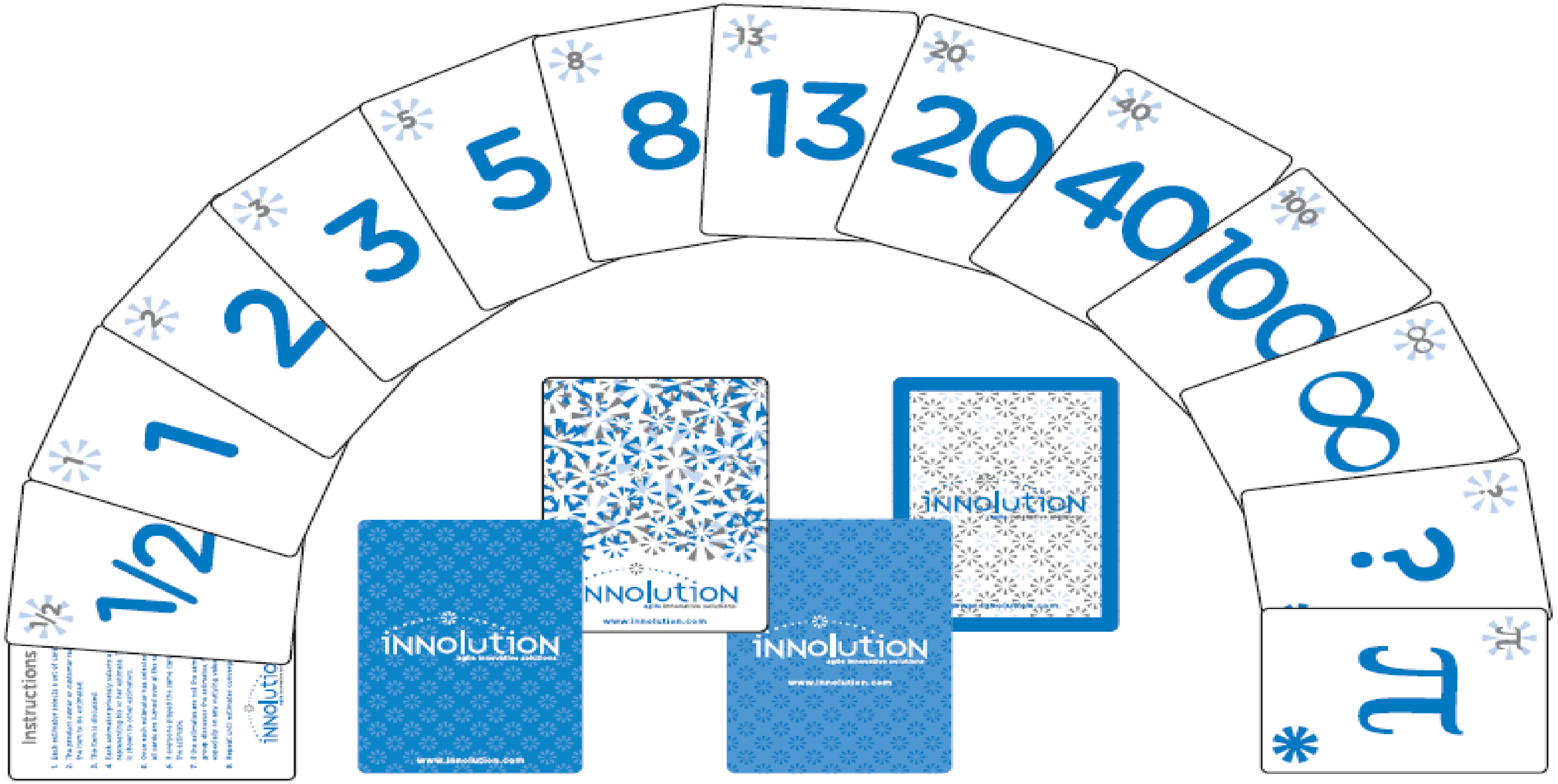
Card	Interpretation
5, 8, 13	Used to size medium items. For many teams, an item of size 13 would be the largest they would schedule into a sprint. They would break any item larger than 13 into a set of smaller items.
20, 40	Used to size large items (for example, feature- or theme-level stories).
100	Either a very large feature or an epic.

Estimation

- **The rules:**
- The rules of Planning Poker are as follows:
- The product owner selects a user story to be estimated and reads the item to the team.
- Development team members discuss the item and ask clarifying questions to the product owner, who answers the questions.
- Each estimator privately selects a card representing his estimate.
- Once each estimator has made a private selection, all private estimates are simultaneously exposed to all estimators.
- If everyone selects the same card, we have consensus, and that consensus number becomes the user story estimate.
- If the estimates are not the same, the team members engage in a focused discussion to expose assumptions and misunderstandings. Typically we start by asking the high and low estimators to explain or justify their estimates.
- After the discussion, we return to step 3 and repeat until consensus is reached.

Estimation

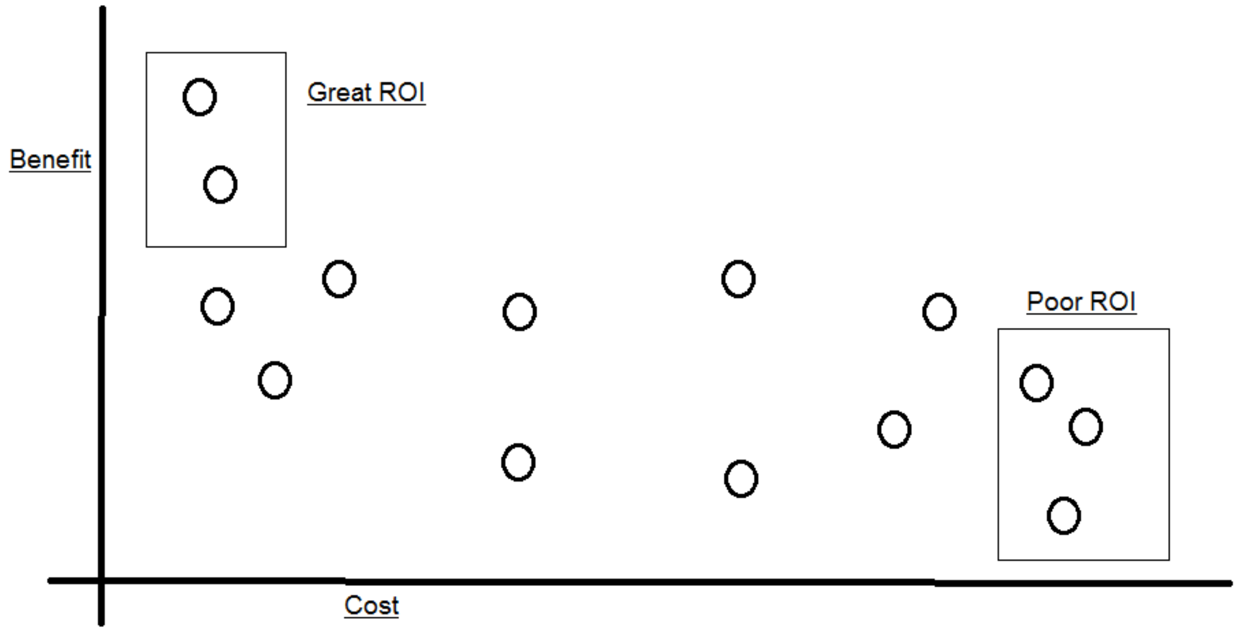
Reference of image: Book-- Essential Scrum—Kenneth S. Rubin



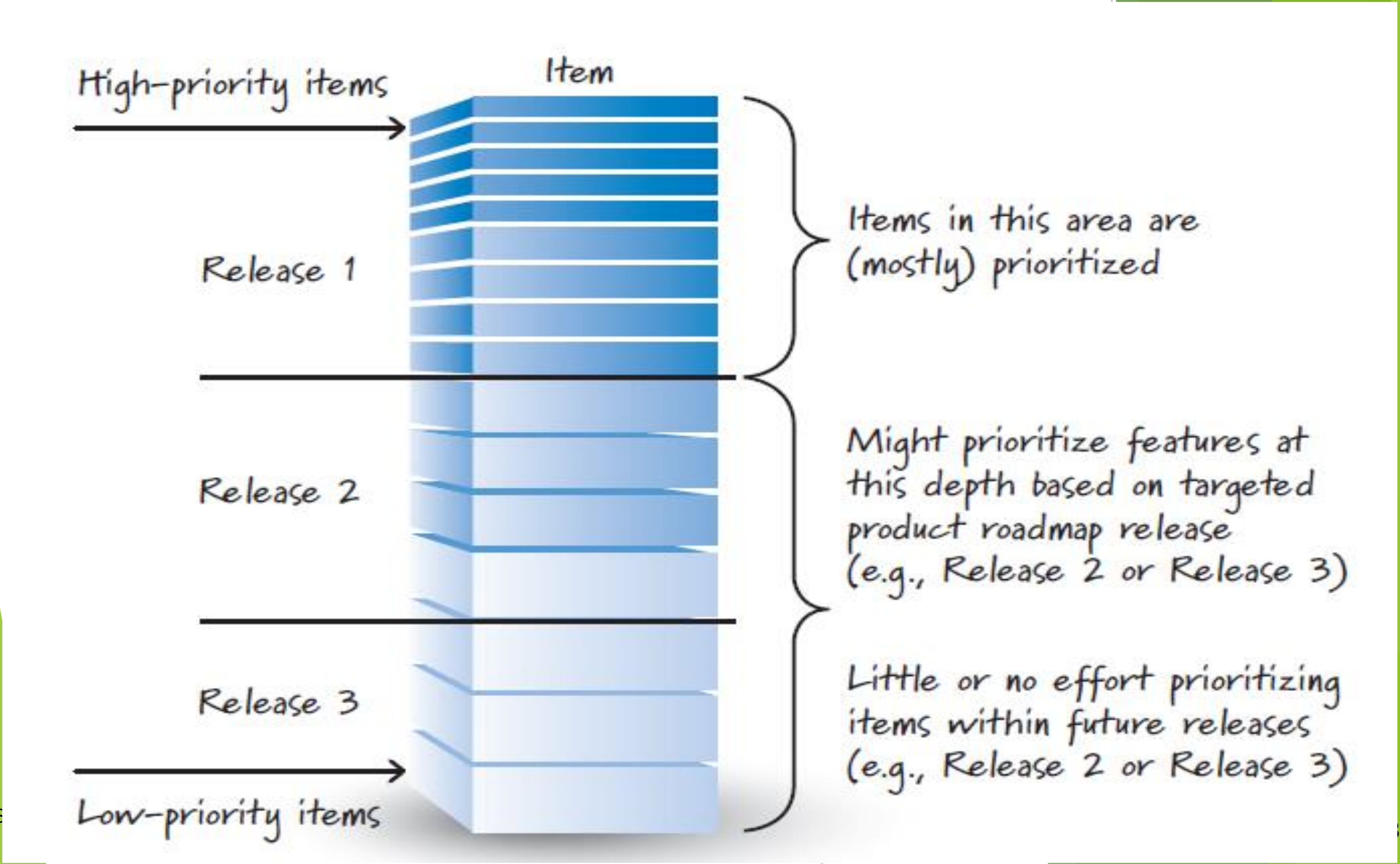
- **Infinity--**Used to indicate that the item is so large it doesn't even make sense to put a number on it.
- ?-- Indicates that a team member doesn't understand the item and is asking the product owner to provide additional clarification.
- Pi---In this context, π doesn't mean 3.1415926! Instead, the pi card is used when a team member wants to say, "I'm tired and hungry and I want to get some pie!"

Priority of user story

- Not all user stories have same benefit
- Some are very important, some are less, but nice to have
- Not all user stories have same cost, some are costly and some are easy to build, thereby less expensive
- By looking at its benefit from business perspective and cost, team decides it return on investment (ROI)
- ROI is simply equal to : benefit / cost



Priority of user stories


High-priority items
Item
Release 1
Release 2
Release 3
Low-priority items
Items in this area are (mostly) prioritized
Might prioritize features at this depth based on targeted product roadmap release (e.g., Release 2 or Release 3)
Little or no effort prioritizing items within future releases (e.g., Release 2 or Release 3)

User stories as Product Backlog Items (PBIs)

Product Backlog Items (PBIs)/ Description of PBI	Estimate	Priority
As a user I want to create a profile so that I want to see my past purchase history	40 (story points)	9
As a customer I want to be able to add items in the shopping cart so that I can buy multiple items	20	9
As a customer I want to search items so that I can see the detail of product, price and buy the better one.	13	9
…	40	7
…	100	5

Title of PBI can be placed in a separate column, AC can be placed in a separate column, Benefit (business value, ROI) can be placed in separate column , Release number can be placed in a separate column

Summary

- Analysis modeling transforms the requirements into a form that naturally leads to the design.
- Requirement analysis
  - Is the base of system analysis
  - Common for us to utilize user hierarchy and use-cases
- Analysis models
  - Content model
  - Interaction model
    - Use-cases
    - Sequence diagrams
    - State diagrams
    - User interface prototype
  - Functional model
  - Configuration model
- Relationship-navigation analysis – content objects and functions!

Readings

R. S. Pressman and D. Lowe: *Web Engineering, A Practitioner's Approach,* McGraw-Hill, 2009.

- Chapter 7: Analysis Modeling for WebApps

(concentrate on the topics covered in the lecture)

Priestley: *Practical Object-Oriented Design With UML, 2ed*.

- Chapter 10: Statecharts

Papers and other reading materials in "Week 4 Readings" folder on CloudDeakin.

Essential Scrum—Kenneth S. Rubin

The agile age-Managing projects effectively using agile scrum---Brian vanderjack