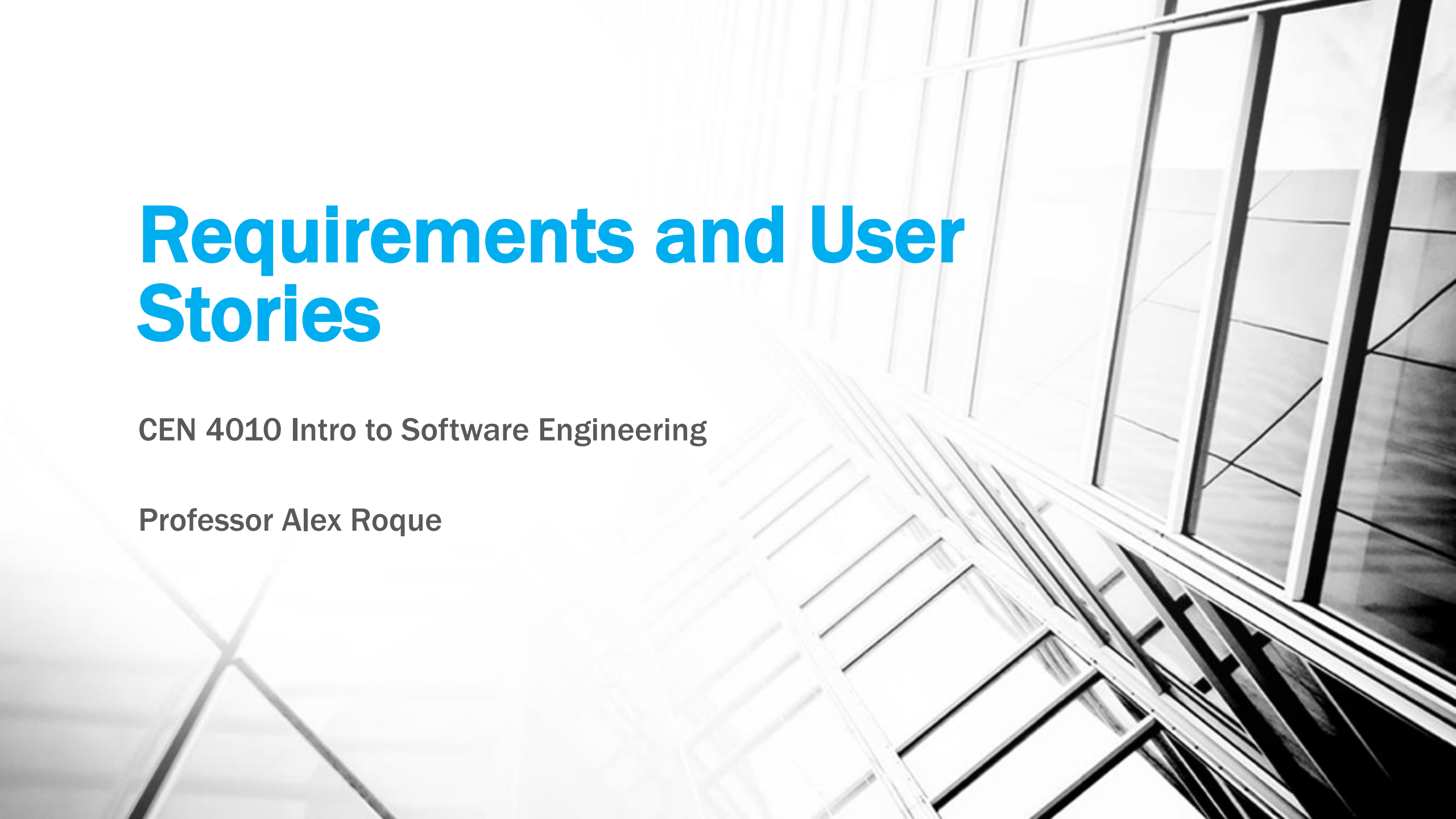
Requirements and User Stories

CEN 4010 Intro to Software Engineering

Professor Alex Roque



Sequential Requirements

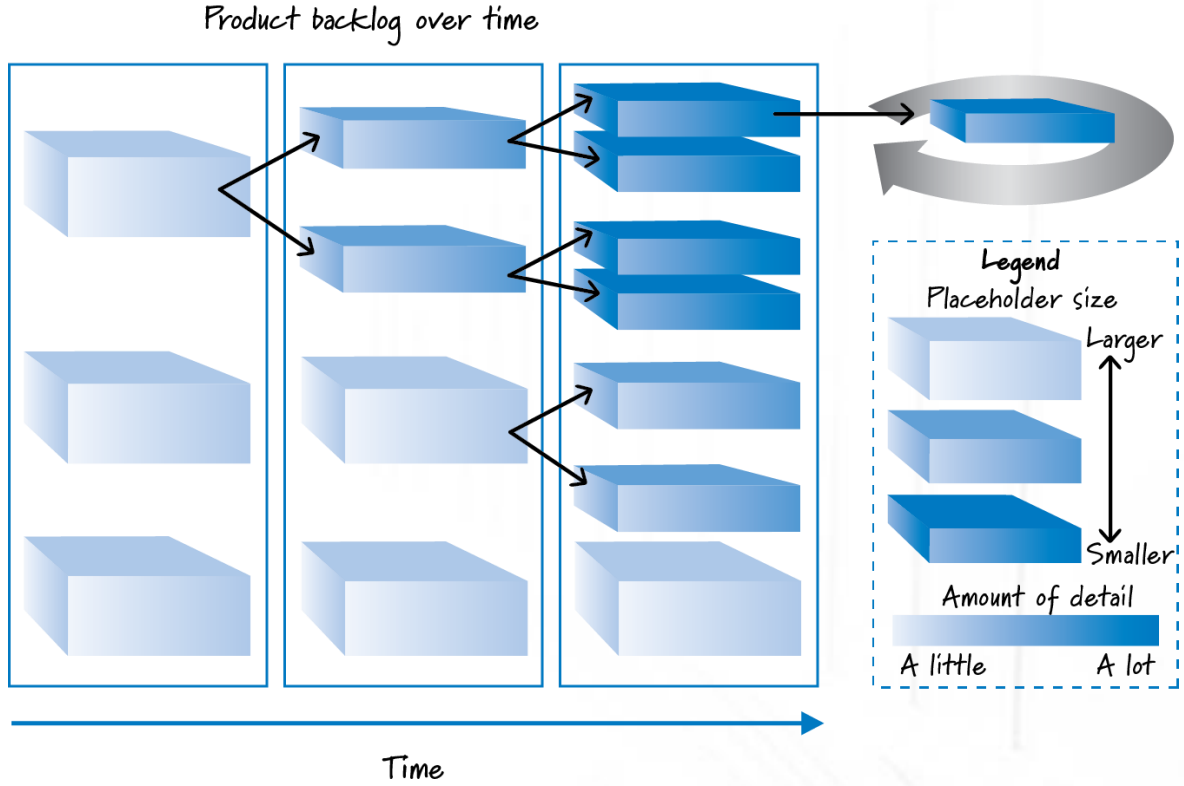
- In Sequential product development, we attempt to provide a specific detail description of what the software needs to do.
- During the analysis phase, business analyst engage in interviewing subject matter experts to create what is know as a Software Requirements Specification Document (SRS).
- Once these requirements are detailed, we try not to do any changes, because any change will need to be driven by a change request process.

Sequential Requirements

- In scrum, we allow a degree of freedom that we can manipulate to meet our business goals, so we can potentially drop requirements if we discover that value of implementing the requirement drops during the sprint.
- When dealing with complex software, attempting to get everything right up front is a difficult is a fool's errand.
- Innovative product development will always have change within it and Scrum can deal with it effectively.

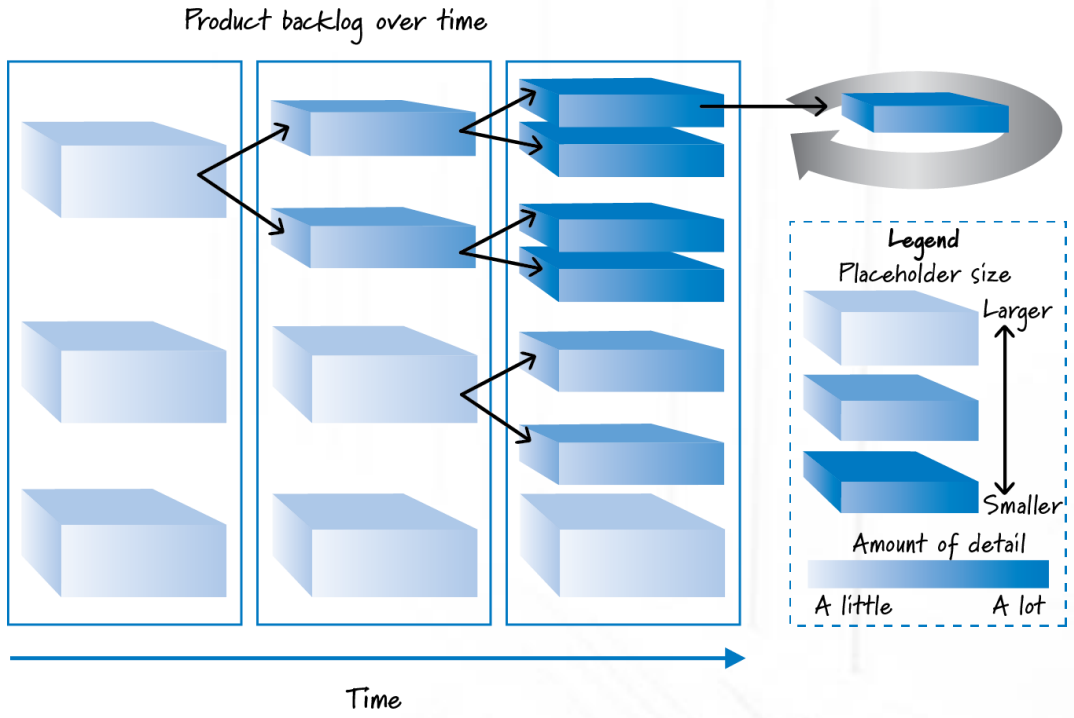
Product Backlog Items

- Scrum creates placeholders for requirements, called **Product Backlog Items (PBIs),** with desired business value
- Requirements in the Product Backlog (PBIs) granularity evolves from large to small as the PBIs approach a Sprint



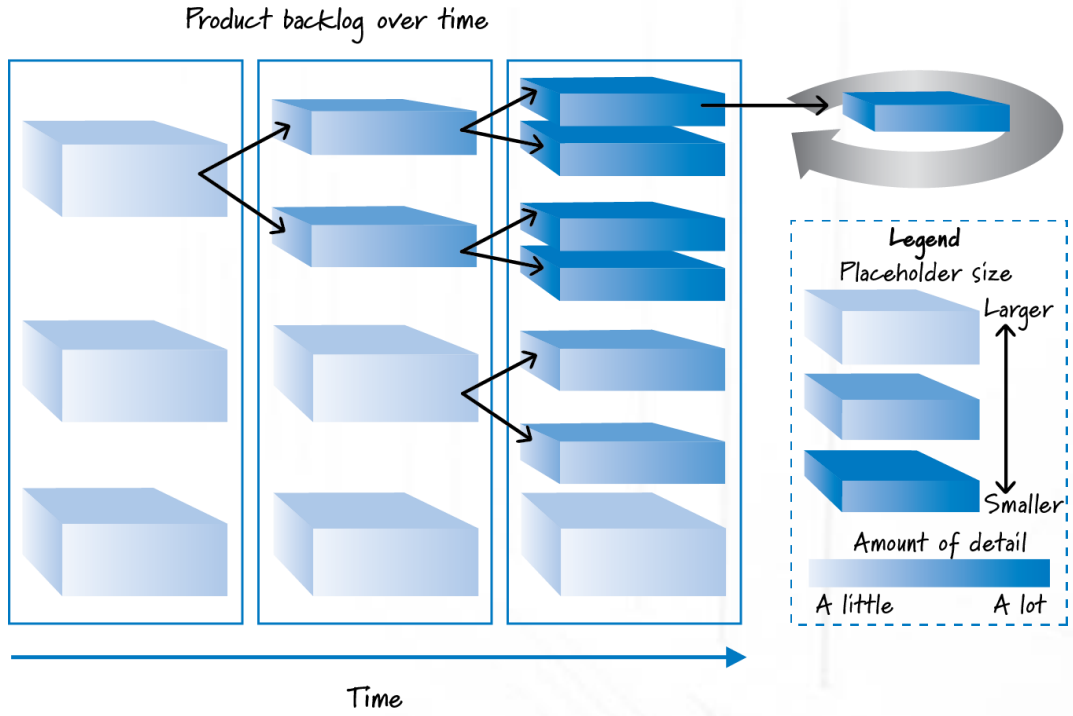
Using Conversations for Eliciting Requirements

- Sequential (Waterfall) development relies heavily on written requirements
  - Look impressive
  - Easily misunderstood
  - Large amounts of time are spent to gather requirements that may change or evolve.



Using Conversations for Eliciting Requirements

- Scrum utilizes to ensure that requirement **progressive refinement** and **bidirectional conversations** are discussed and communicated
  - Level of detail during conversation changes as a requirement (PBI) gets closer to a Sprint
- PBIs can be gathered and put into a formal document if needed



User Stories

- Scrum framework does not dictate a specific artifact to use for documenting requirements/features/etc.
- User Stories are widely used to document many types of requirements capturing **business value** in the process
- Sometimes other artifact type can be used, **for example**, to simply reference a defect in the defect-tracking system rather than writing a user story about the defect

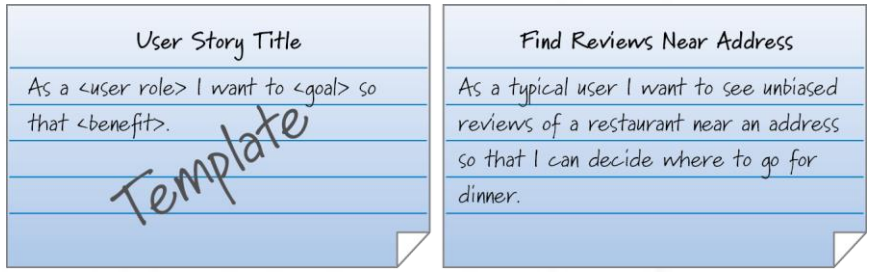
Typical User Story Format: (Card)

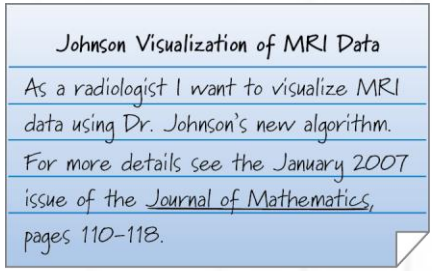
As a User Role **, I want to**

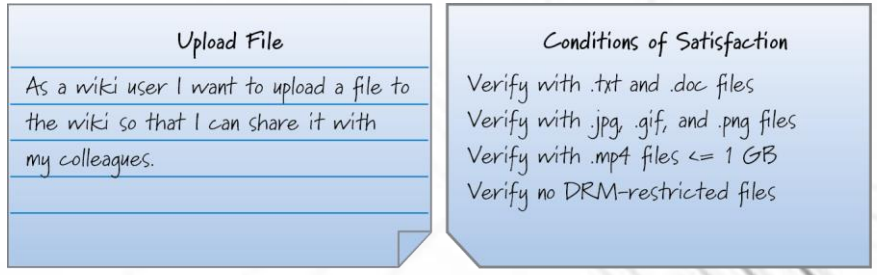
Goal **so that** Benefit **.**

User Story Artifacts

- Ron Jefferies (2001) coined the 3 c's: **card**, **conversation**, and **confirmation**
- Card:
  - We used (and still do) **3x5 cards** and **Sticky Notes** (Automated tools simulate these – e.g., Trello)
  - More detail on the **backside**
- Conversation:
  - Includes Dev. Team and Stakeholders
  - Usually **on-going** rather than one time
  - Can result in other **supporting artifacts**, such as UI sketches, elaboration of business rules

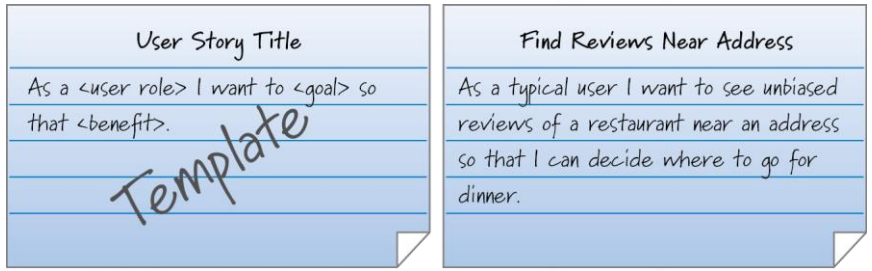


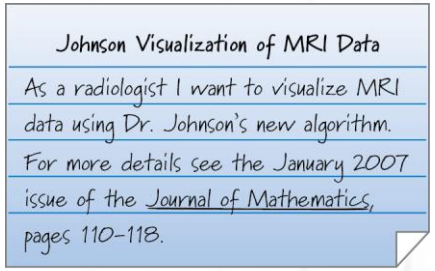


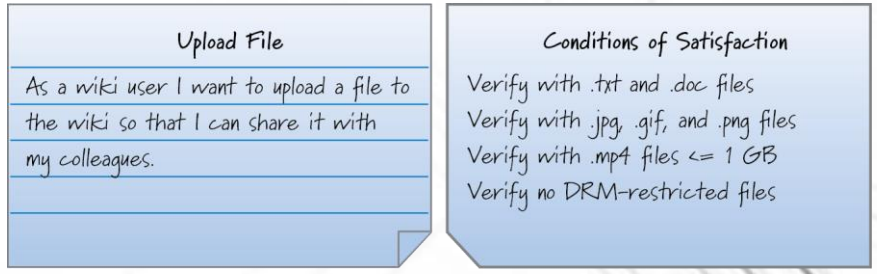


User Story Artifacts

- Confirmation
  - Conditions of satisfaction, **acceptance criteria**
  - Dev team will probably have **many more "tests"** that are not part of the PO's acceptance criteria







User Story versus Predefined Requirement

- Remember the Agile Manifesto?
  - We value “Individual and Interactions” over “Processes and Tools”?
- The purpose of writing a story is to foster communication and collaboration to create functionality that brings value to the customer.
- No more finger pointing:
  - Developer - “that wasn’t specified in the requirement”
- Again, we know change can take place and we embrace it, as long its not during the execution of the sprint (even then slight modifications are allowed).

Being Agile to Create Complex Software

- Think about the statement ““that wasn’t specified in the requirement”
- Why is that considered counter-intuitive to the agile manifesto?
- In a very competitive enterprise landscape, what does this do to customer loyalty?

User Story – Level of Detail

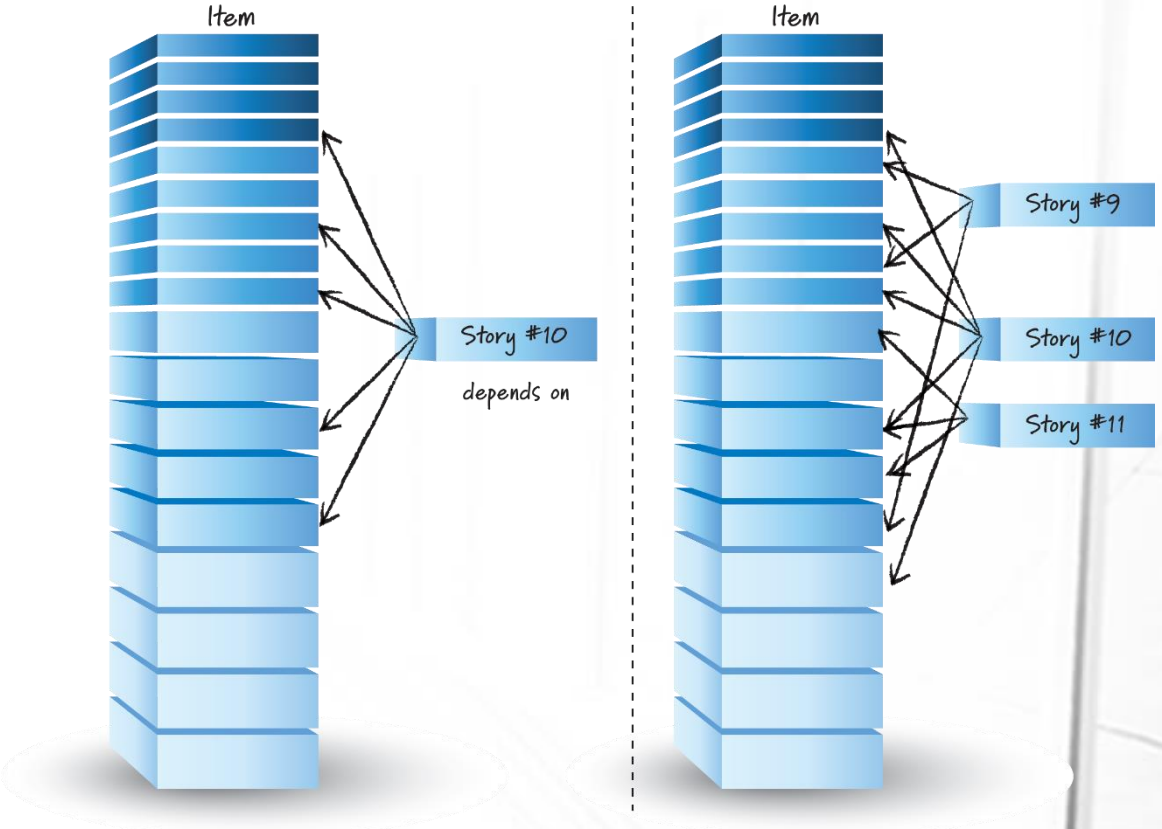
- The work that needs to be done can be broken down into different levels:
  - **Epic:** An Epic is large story that would take several months to implement. Also it is possible that the Epic would need further work to be clearly defined.
    - **Feature:** The next level below an epic can take multiple sprints to implement thus need multiple weeks to implement.
      - **User Story:** A Sprint-able user story can be estimated in days and can be implemented within a sprint.
        - **Task:** A story can be broken down into tasks by the team. Each task is usually estimated in hours and assigned to one individual.
- Theme: A theme can group a series of related user stories.
  - Example: User stories 001, 002 and 003 have a theme of creating the "Integration APIs"

User Stories – INVEST

- **INVEST is an acronym used to describe that User Stories should be**
  - Independent
  - Negotiable
  - Valuable
  - Estimatable
  - Small
  - Testable

User Stories – INVEST

- Independent
  - Stand alone or loosely coupled to others
  - Interdependencies complicate estimating, prioritizing, and planning
- Negotiable
  - Discuss What and Why, not How
- Valuable
  - To user or customer
  - Okay to have some technical stories so PO understands what it being paid for
- Estimatable: Should be complete in sprint
- Small (sized appropriately)
- Testable : Should have an acceptance criteria



Nonfunctional Requirements / Knowledge Acquisition

- Nonfunctional Requirements:
  - Represent system level constraints
  - They can be written as independent stories or can be an acceptance criteria as part of another story
  - Its important to do test nonfunctional requirements during the sprint, instead of leaving it until later.
  - An organization may have feature standards for nonfunctional requirements, such as:
    1. Client request to our application should respond in less than 2 seconds.
    2. Messages received by our native app should have a pop-up notification.
    3. Application must support Firefox and Chrome.
- These NFRs might be tied to a service level agreement (SLA).

Nonfunctional Requirements / Knowledge Acquisition

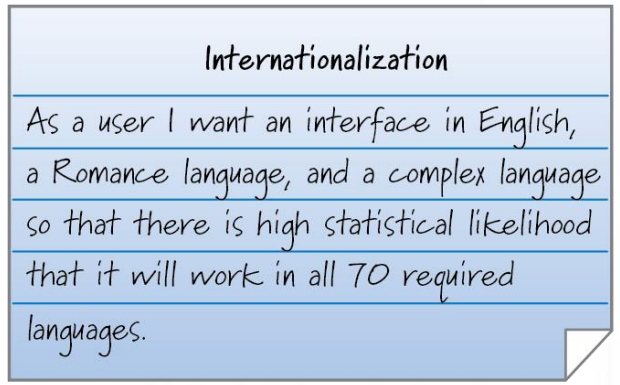
- Knowledge Acquisition:
  - These stories contain exploration that are needed for decision making purposes:
    - Prototype, proof of concept, experiment, study, spikes
  - The product owner must be able ascertain the economic value of these stories (will working on POC for a sprint have a cost that is less than the alternative of not working on the POC?)

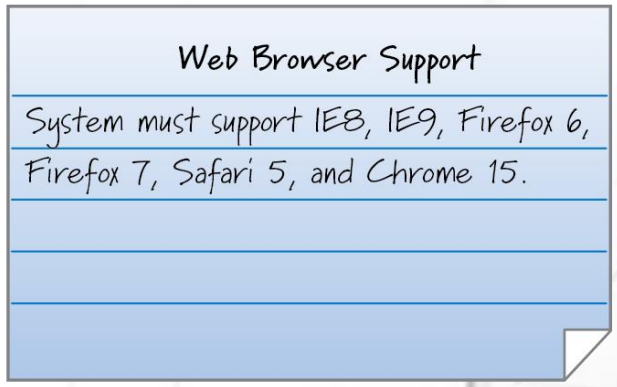
Knowledge Acquisition- Research Spike

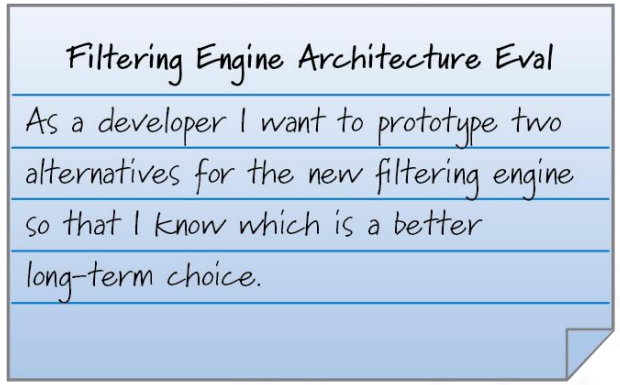
- A research spike or a spike is a specific user story that is not meant to provide direct value to the business but is instead used to research or to explore an unknown tied to the features being developed.
- Its purpose is to perform exploratory work and provide the result of that research as the outcome of the user story.
- The output of the user story can be:
  - Proof of Concept Implementation
  - Research document
  - Completed Implementation
  - Partial work that will be consumed as a dependency for future user stories

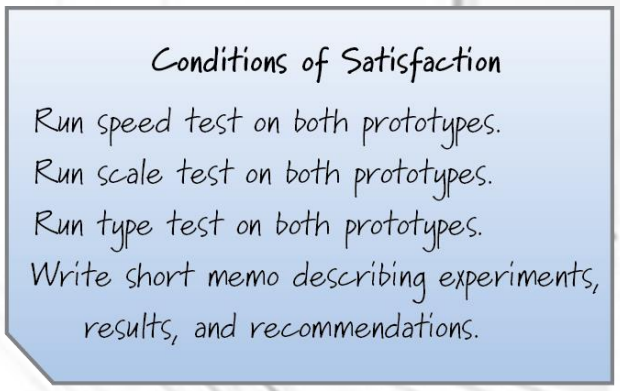
Nonfunctional Requirements / Knowledge Acquisition

- **Nonfunctional Requirements** represent system-level constraints
  - Can be written in a non-user story way
  - Good idea to include as part of **Done**
- **Knowledge-Acquisition Stories**
  - Exploration includes prototyping, proof of concept, experiment, study, etc.
  - Cost benefit approach



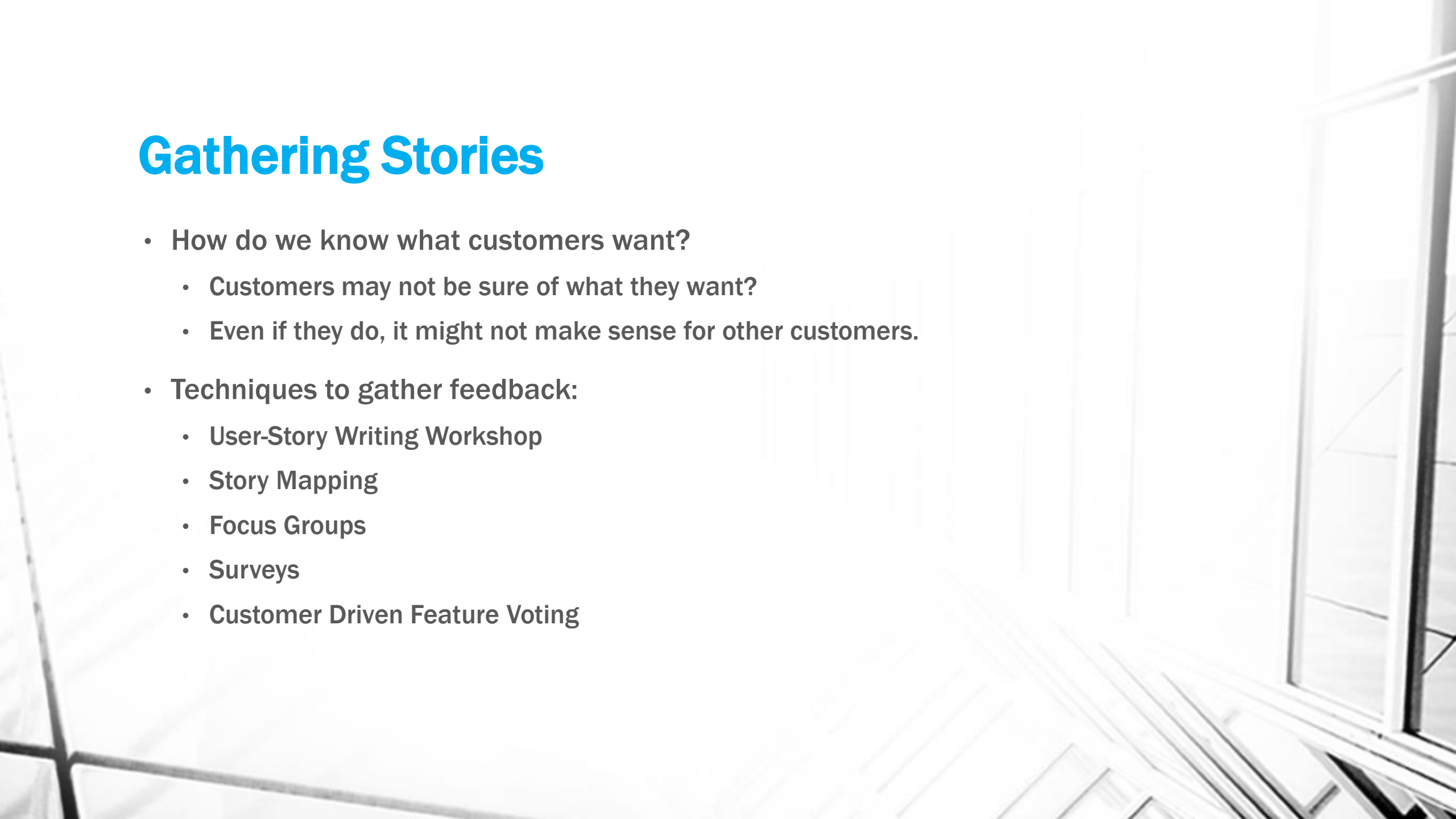






Gathering Stories

- How do we know what customers want?
  - Customers may not be sure of what they want?
  - Even if they do, it might not make sense for other customers.
- Techniques to gather feedback:
  - User-Story Writing Workshop
  - Story Mapping
  - Focus Groups
  - Surveys
  - Customer Driven Feature Voting



User Story Writing Workshop

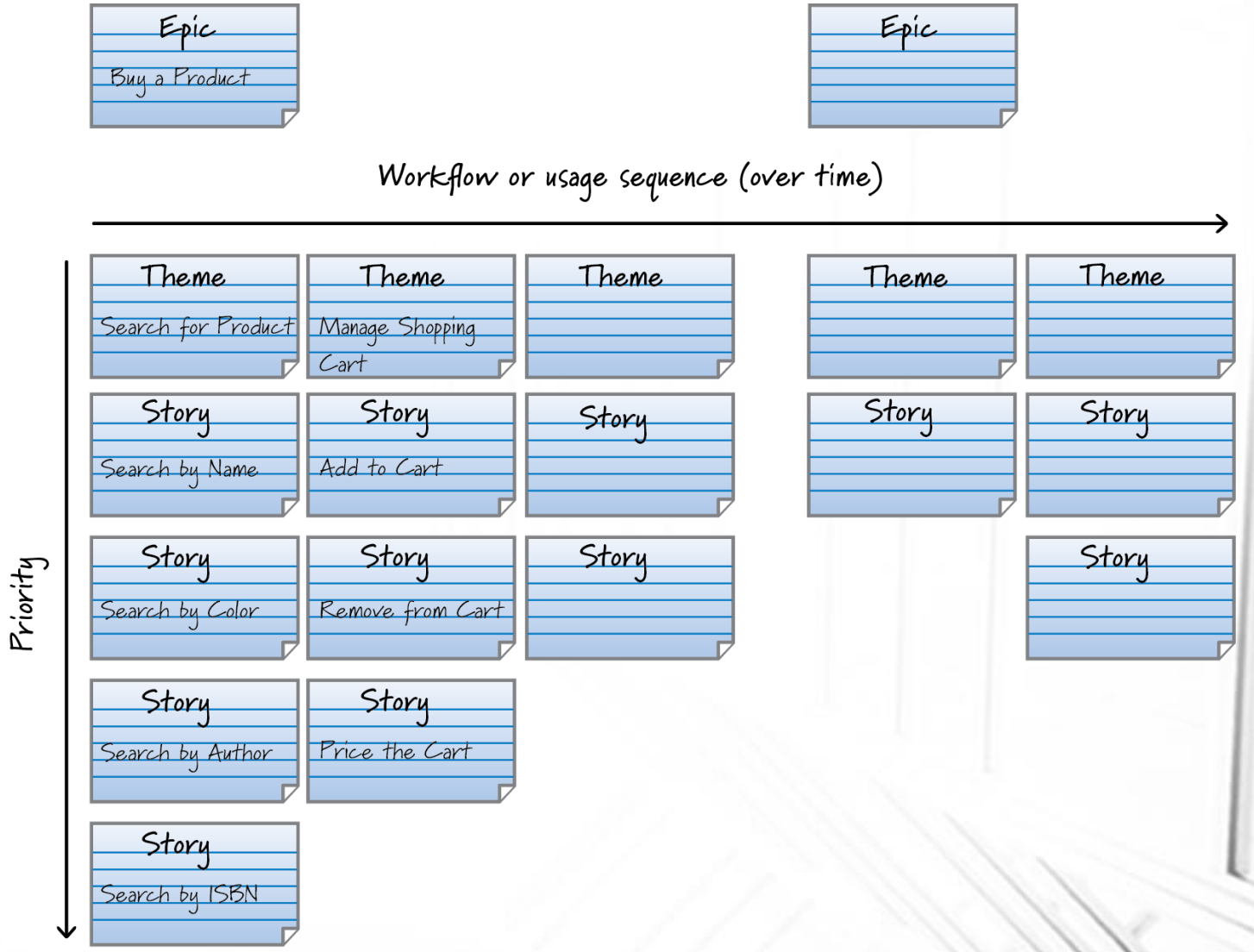
- A User Story writing workshop is a collective brainstorm to obtain the desired business value and user story placeholders.
- Usually composed of the Product Owner, Scrummaster, development team, and internal/external stakeholders.
- Personas are often defined. Personas are fictional users that represent a real-life user from the system.
  - Robert Shopper: Robert Shopper is a user that browses the bookstore and add books to the cart by browsing.
- By understanding the persona, you can add features that make sense for the specific user type.

User Story Writing Workshop

- What methods can we use to determine the details needed in a user story?
- Method 1:
  - Top-Down: Start at the Epic level and refine into user stories.
- Method 2:
  - Bottom-Up: Start with specific user stories and synthesize the overall value.
- Be careful…make sure that you can always the work back to an epic or theme.

Story Mapping

- Start with an Epic and identify the key themes to the usage of that Epic
- User-centric approach to generating a set of user stories
- Decompose a user activity into a workflow for further decomposition
- Patton used terms **activity**, **task**, and **subtask**



User Story Components

- A User Story Usually contains the following components:
- Title: The User Story Title typically has a description of what the user story

  <User Story Title>

  As a <user role/persona> I want to <goal> so that <benefit>

  <Details>

  - Goal: This is the what. What is that the aggregate work of the team will allow the product to do.
  - Benefit: This is the why. We detail here why is beneficial to implement the work.

User Story Components

- In addition, a User Story usually contains the following components:
- Acceptance Criteria:
  - These are the functional or non-functional requirements that need to be met to mark the user story as complete.
- Acceptance Tests:
  - These are the tests that need to be executed to make sure that the user story can be completed.
  - They can be created from reviewing the acceptance criteria or can be in addition to the acceptance criteria expectations.
  - They can be created by any member of the team (engineers or tester) with guidance from the product owner as well.
- Estimate: Can be in man hours or relative story points.

User Story details

- Consider yourself part of a scrum team. Why would one of these User Story items resonate with you and why?
  - *<User Story Title>*
  - *As a <user role/persona> I want to <goal> so that <benefit>*
  - *<Details:>*
  - *Acceptance Criteria*
  - *Acceptance Test*
- *Is there anything else you feel would be important to add?*