

# User Stories – The Art of Writing Agile Requirements

# Agenda

- Introduction
- Overview of Agile/Scrum
- From Vision to Acceptance Criteria
  - Modeling Users & Customers
  - Epics, Features & User Stories
  - Elaborating from Vision to Story
  - Acceptance Criteria & Testable Examples
- Q & A

---

# Overview

## Agile & Scrum

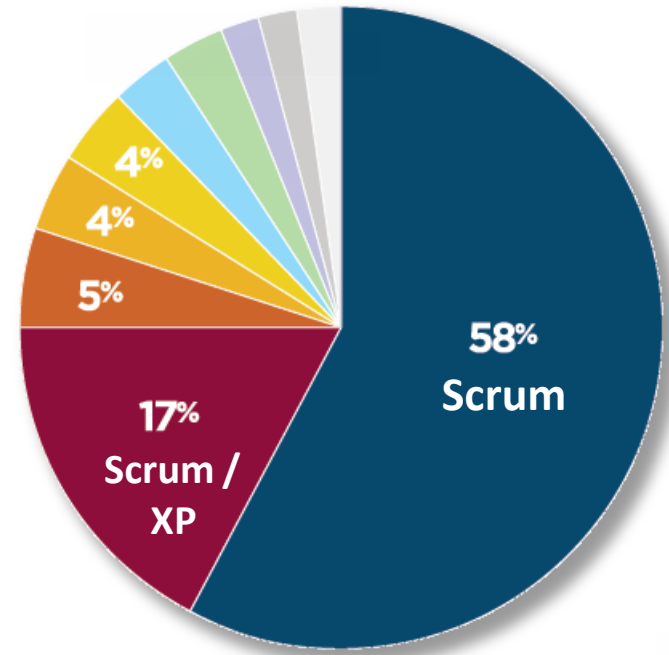
---

# The Agile Landscape

“Agile” describes a number of related methods.

**Scrum is the most popular.**

- **Scrum**  
*Jeff Sutherland & Ken Schwaber*
- **Extreme Programming (XP)**  
*Kent Beck, Ward Cunningham, Ron Jeffries*
- **Kanban**  
*David Anderson*
- **Scaled Agile Framework (SAFe)**  
*Dean Leffingwell*



Source: 2010 State of Agile Development Survey, VersionOne

# Dealing with Uncertainty

You don't need agile if you know what to build, who to build it for, and how to build it



Initial Plan

Use agile when you have uncertainty...

Empirical methods monitor progress & direct adaptation



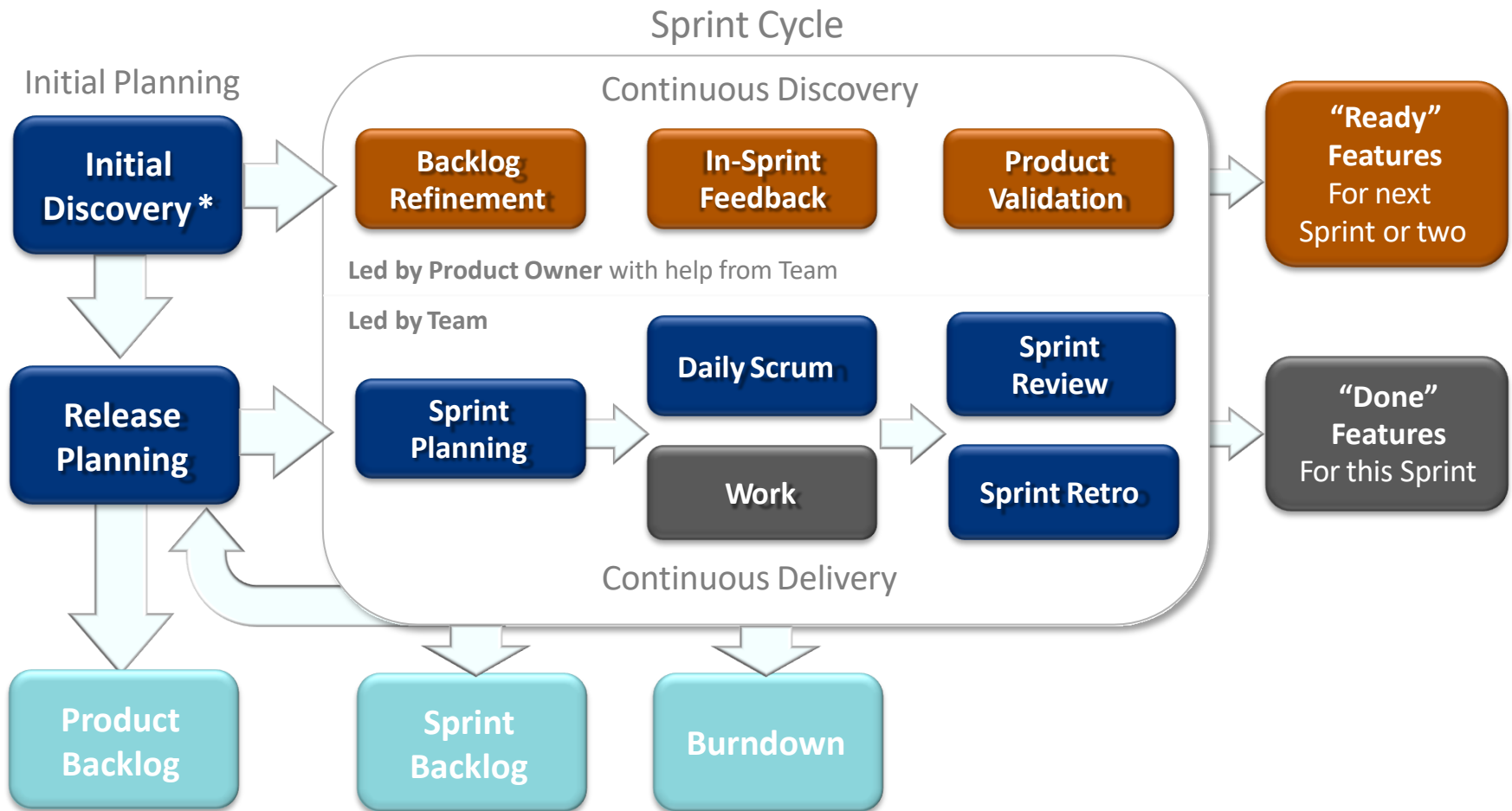
Better Plan

**What to build:** End Uncertainty

**How to build it:** Means Uncertainty

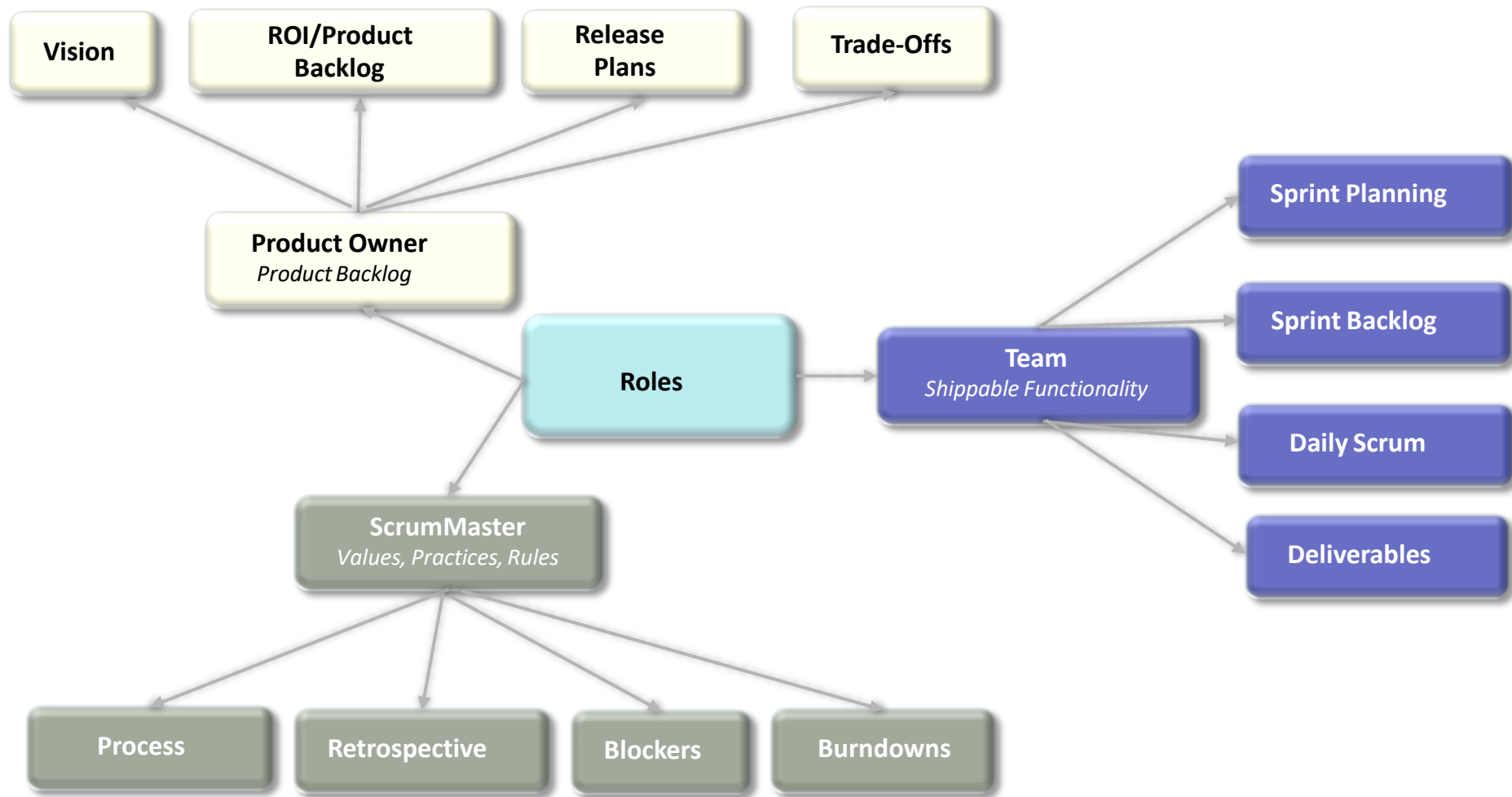
**Who to build it for:** User Uncertainty

# Ceremonies and Artifacts of Scrum



\* Discovery is not explicitly part of the Scrum Framework

# Roles



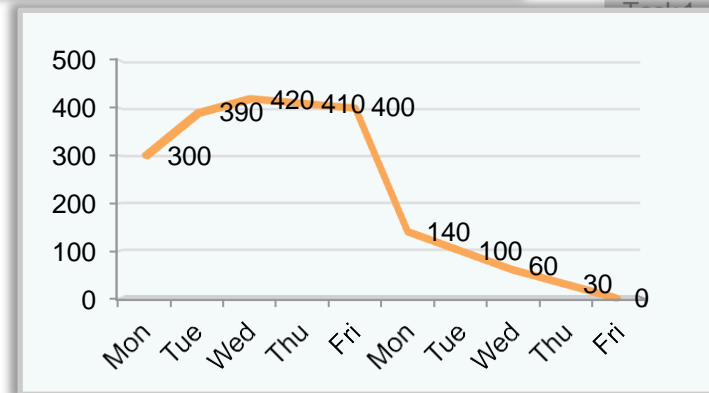
# Tools

- **Product Backlog**  
Prioritized list of all items (PBI) required to launch a successful product
- **Sprint Backlog**  
Tasks to get committed PBIs to done within Sprint
- **Task Board**  
Stories and tasks for the Sprint tracked from start to completion
- **Burndown/burn-up Chart**  
Visual aid for tracking team progress and forecasting expected completion dates

PB Item	Priority	Size
User Story	High	2
User Story	High	3
User Story	High	5
User Story	Medium	2
		13

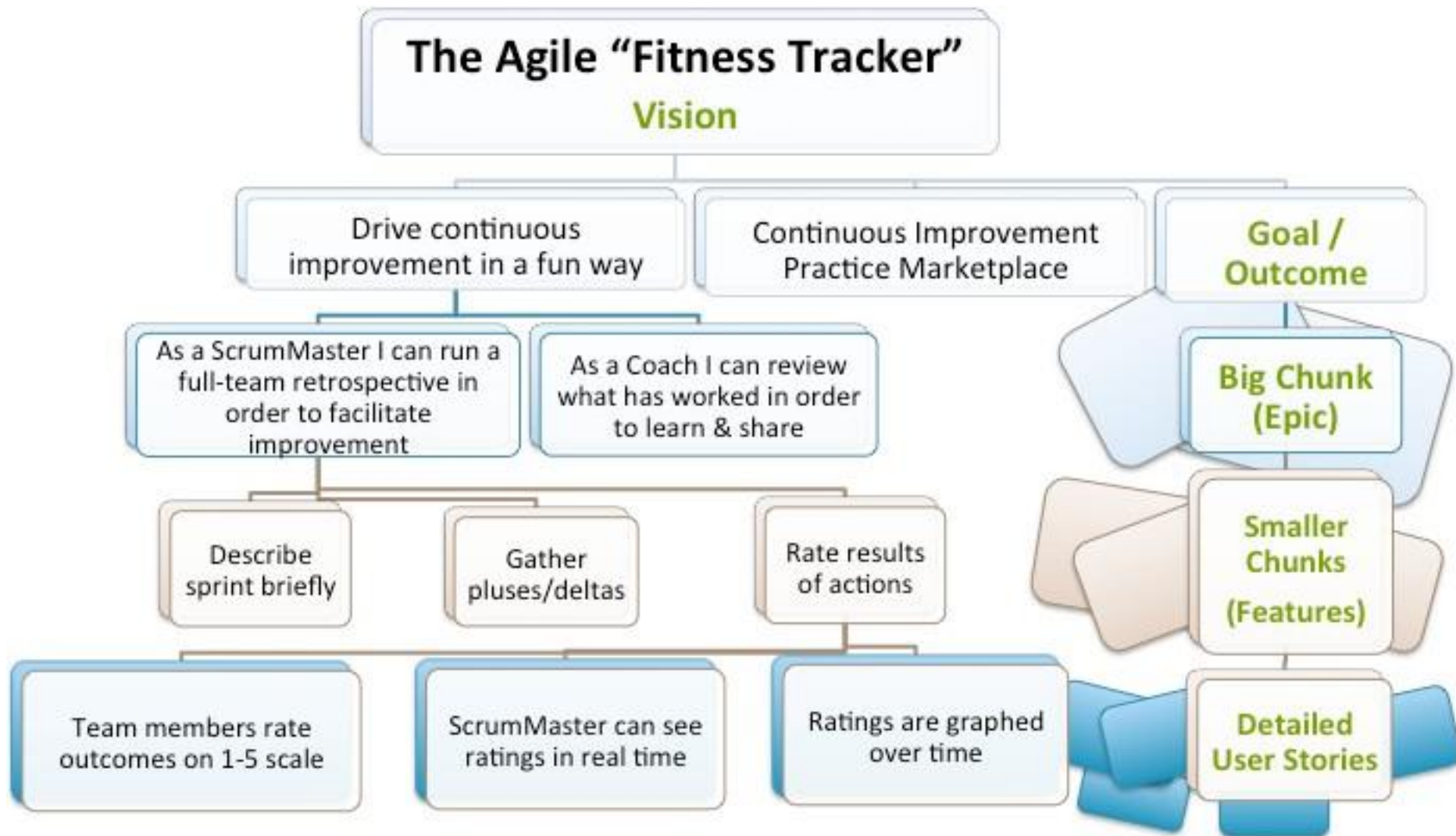


SB Item	Priority
User Story	High
Task 1	
Task 2	
Task 3	
User Story	Medium
Task 4	
Task 5	
Task 6	
Task 7	
Task 8	
Task 9	
Task 10	
Task 11	
Task 12	
Task 13	
Task 14	
Task 15	
Task 16	
Task 17	
Task 18	
Task 19	
Task 20	
Task 21	
Task 22	
Task 23	
Task 24	
Task 25	
Task 26	
Task 27	
Task 28	
Task 29	
Task 30	
Task 31	
Task 32	
Task 33	
Task 34	
Task 35	
Task 36	
Task 37	
Task 38	
Task 39	
Task 40	
Task 41	
Task 42	
Task 43	
Task 44	
Task 45	
Task 46	
Task 47	
Task 48	
Task 49	
Task 50	
Task 51	
Task 52	
Task 53	
Task 54	
Task 55	
Task 56	
Task 57	
Task 58	
Task 59	
Task 60	
Task 61	
Task 62	
Task 63	
Task 64	
Task 65	
Task 66	
Task 67	
Task 68	
Task 69	
Task 70	
Task 71	
Task 72	
Task 73	
Task 74	
Task 75	
Task 76	
Task 77	
Task 78	
Task 79	
Task 80	
Task 81	
Task 82	
Task 83	
Task 84	
Task 85	
Task 86	
Task 87	
Task 88	
Task 89	
Task 90	
Task 91	
Task 92	
Task 93	
Task 94	
Task 95	
Task 96	
Task 97	
Task 98	
Task 99	
Task 100	





# The Big Picture



---

# Vision

## Aligning Goals & Constraints

---

# Crafting a Vision Statement

For **Target Customers**  
Who **Statement of Need**  
The **Product Name**  
Is a **Category**  
That **Compelling Reason to Buy & Use**  
Unlike **Competition / Alternative**  
Our Product **Differentiator**



*As described by Geoffrey Moore in **Crossing the Chasm**  
(Thanks to Gabrielle Benefield for the reference)*

---

# Simulation

## Restaurant Finder

---

# Exercise – Prepare a Pitch

## Create a Vision Poster for your simulation project with:

1. A product name;
2. A product logo;
3. A product slogan or jingle; and
4. Three (3) compelling reasons to buy your product.

---

# Personas

## Customer & User Modeling

---

# Users vs. Customers



## **Users** interact directly with the system

They are important to understand, because:

- Knowledge of current usage patterns helps to design better, more usable systems.
- Unsatisfied users will work around the system, nullifying its advantages and eventually eliminating it.



## **Customers (sponsors)** make buying / adoption decisions

They are also important, because:

- They have their own wish lists that may have little to do with their users' needs.
- They make the purchasing decisions, so if they aren't happy, you won't get in the door.

# Looking Across Usage Scenarios

- **Personas** represent a type of user across usage contexts.
  - One member of our current or desired audience in a tangible, less ambiguous way.
  - Provide a name, a face, and a description giving us a mental model of our users allowing us to emphasize with them and **predict how they will use our software.**
- **Level of detail**
  - Add just enough detail to aid empathy, more details can be distracting.
  - Lightweight personas will suffice for many.





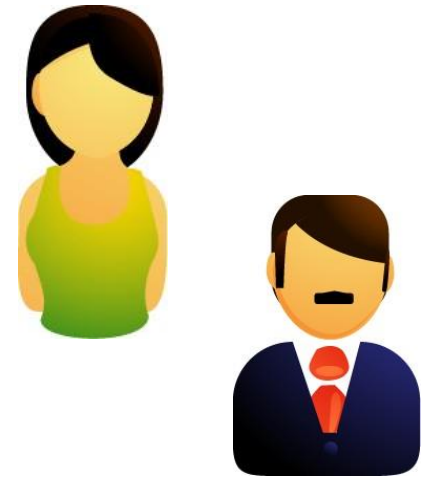
# User Models Summary

- **Use what works** user roles, personas, etc., without getting hung up in vocabulary.
- **Prioritize your user(s)** and prioritize stories for them.
- Post **big charts** (e.g. personas) in team room to aid **empathy**.
- **Focus testing and evaluation** on the right users, identifying test subjects similar to your models.
- Base models on reality (ethnography / field study):
  - Usability Testing
  - Observation
  - Interviews
  - Data Analysis
  - Feedback Forms
  - Surveys, etc.

# Exercise: Create Personas

## Who are your most critical personas, or early adopters?

1. **List potential stakeholders** that would represent the Customers & Users of your product.
2. **Prioritize these stakeholders** and pick two to elaborate.
3. **Create at least two personas** by writing brief stories that outline the motivations and goals of these customers or users.



---

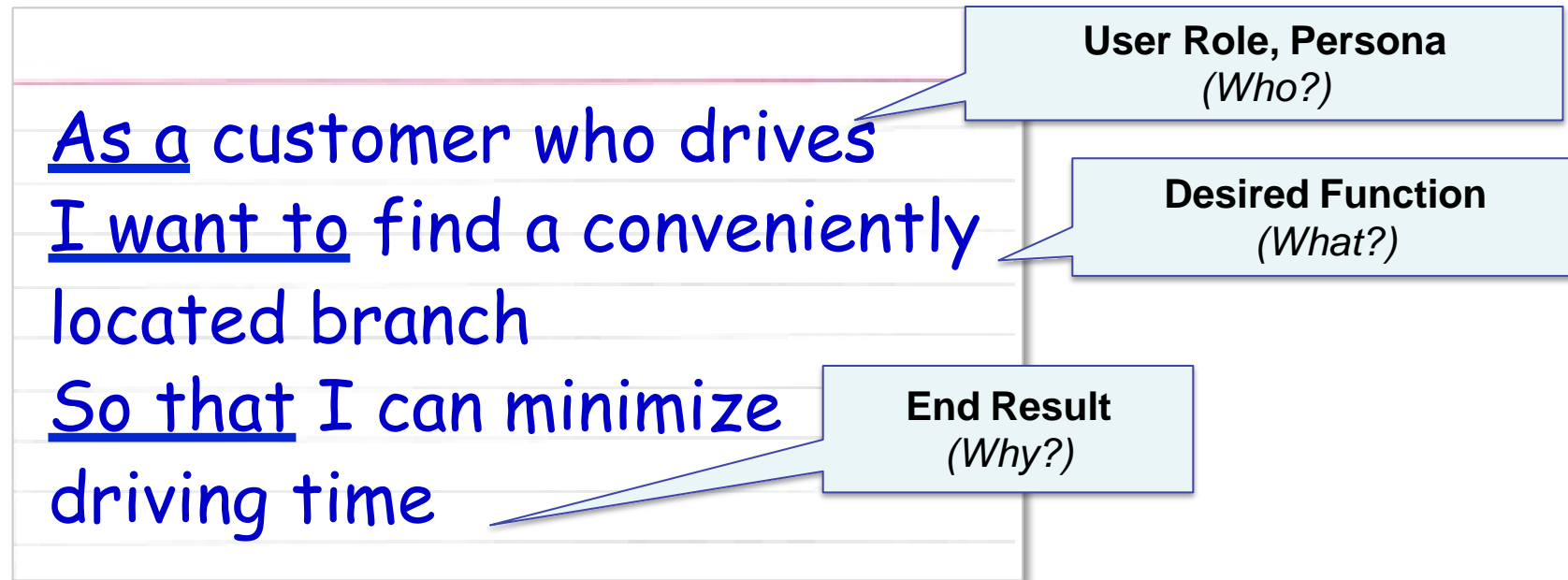
# User Stories & the Backlog

Working with Agile “Requirements”

---

# User Stories

The basic user story template is simplistic, it helps us remember a need while providing context.



What is not specified?

# User Stories at a Glance

## Key Characteristics

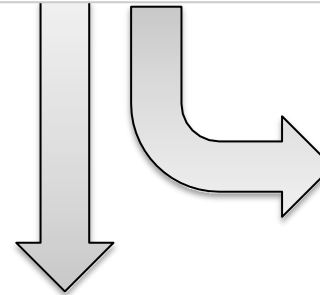
- High-level descriptions of desired functionality and goals
- “**Contracts for conversation**”, not all-inclusive requirements
- Pulled into the **Sprint Backlog** from **Product Backlog**
- Contain **Acceptance Criteria** to define “Done”
- **Vertical slices** of the system’s functionality

### *Product Backlog User*

As a user I want to create an account so that I can shop online.

Estimate 13 Points

Priority 1 (High)



As a user I want to enter my billing information.

Estimate 4 points

Priority 1 (High)

As a user I want to enter my personal information.

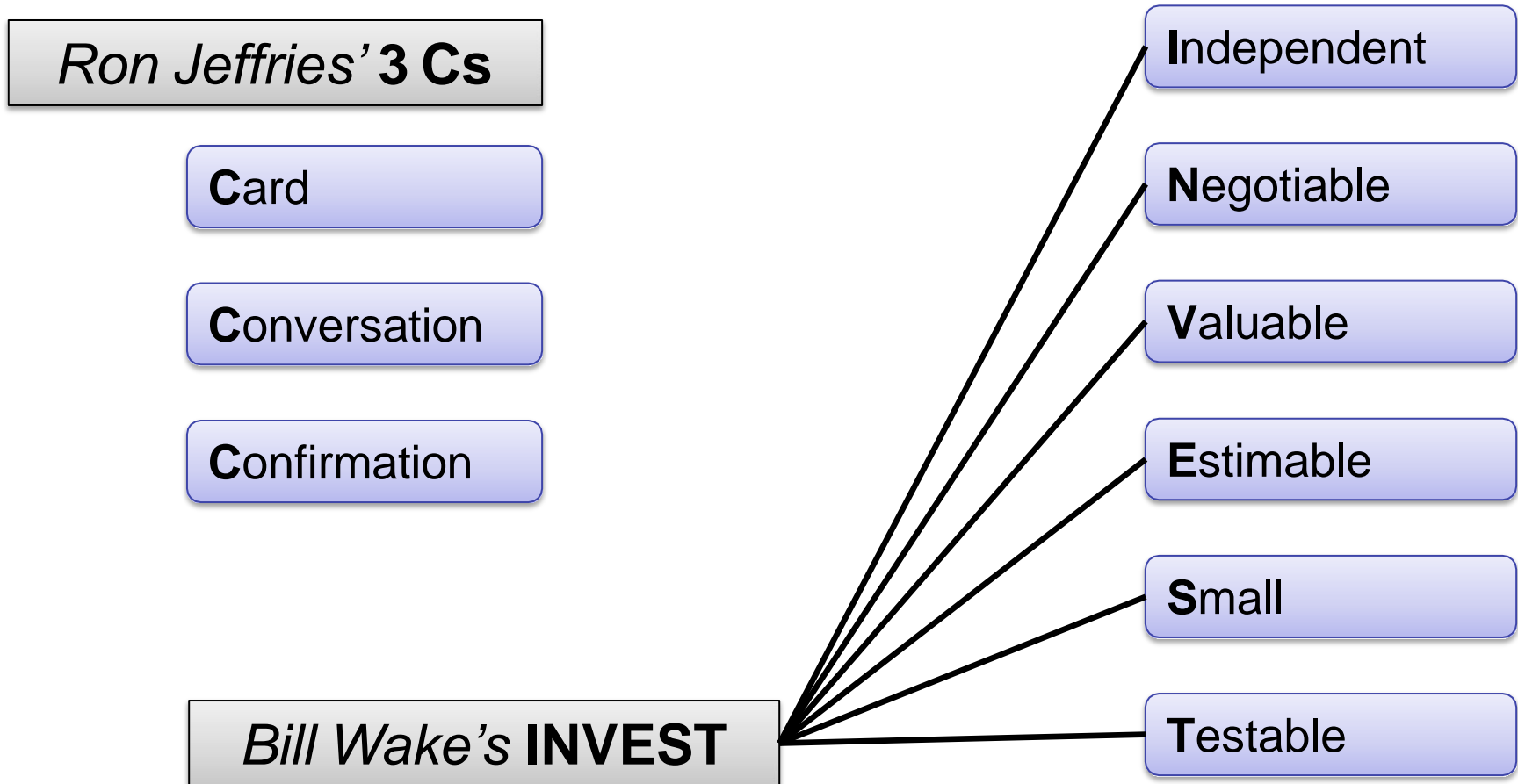
Estimate 4 points

Priority 1 (High)

### *Sprint Backlog User Stories*

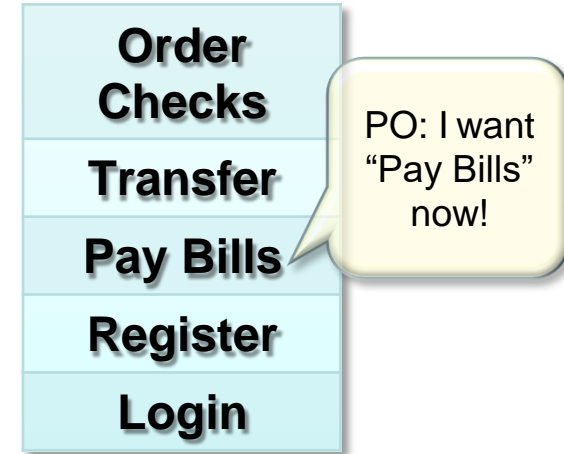
*Work in Agile projects is organized by Units of Value, rather than by Architectural Layer.*

# What Makes a Good Story?



# User Stories: Invest - Independent

- If all stories are independent, any one can be picked and delivered in isolation
- For large systems this is nearly impossible!
- But, minimizing, identifying and prioritizing, dependencies can result in a better backlog
- Which user story must come first?



As prospect  
I want to register  
So that I can  
execute electronic  
transactions

As a user  
I want to pay bills  
online  
So that I don't have  
to write checks

# User Stories: Invest - Negotiable

- Leaving room for give and take and decide the details when you have more context
  - High priorities stories should be more precisely defined
  - Low priority stories should have more play

## Essence today

2012

January

February

March

April

S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S
1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7
8	9	10	11	12	13	14	8	9	10	11	12	13	14	4	5	6	7	8	9	10	11	12	13	14	15	16	17
15	16	17	18	19	20	21	15	16	17	18	19	20	21	12	13	14	15	16	17	18	19	20	21	22	23	24	25
22	23	24	25	26	27	28	19	20	21	22	23	24	25	18	19	20	21	22	23	24	25	26	27	28	29	30	31
29	30	31					26	27	28	29				25	26	27	28	29	30	31							

May

June

July

August

S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S
1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7
8	9	10	11	12	13	14	8	9	10	11	12	13	14	8	9	10	11	12	13	14	15	16	17	18	19	20	21
15	16	17	18	19	20	21	15	16	17	18	19	20	21	12	13	14	15	16	17	18	19	20	21	22	23	24	25
22	23	24	25	26	27	28	19	20	21	22	23	24	25	22	23	24	25	26	27	28	29	30	31				
29	30	31					26	27	28	29	30			29	30	31					28	29	30	31			

September

October

November

S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S
1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7
8	9	10	11	12	13	14	8	9	10	11	12	13	14	8	9	10	11	12	13	14
15	16	17	18	19	20	21	15	16	17	18	19	20	21	15	16	17	18	19	20	21
22	23	24	25	26	27	28	22	23	24	25	26	27	28	18	19	20	21	22	23	24
29	30						29	30	31					25	26	27	28	29	30	

Data  
List

## Details later

## As a driver

I want to get directions to conveniently located stores So that I get there quickly

### Acceptance Criteria:

Show locations on map  
~~Show locations on Google~~  
~~Maps~~

Defer details until you are close to building, in this case update the acceptance criteria



# User Stories: Invest - Valuable

The user story must have value to the user and to the business

As a user

I want to have my  
previous orders stored  
in the database  
So they will be there  
permanently

As a repeat customer

I want to access old orders  
So that I can quickly  
purchase the same products  
again

As a customer

I want 75% off all  
purchases  
So I can save money

There is clearly  
value to the user,  
but is there value  
to the business?



# User Stories: Invest – Estimable/Small

- If you can't estimate it, it is either too large, too vague, too risky, or some combination thereof
- Solutions include adding acceptance criteria, splitting the story, or better defining it

Easier to estimate, perhaps small enough to complete in a few days

As a customer  
I want a self service center  
So that I can address basic needs 24 by 7 by 365 from my computer

Too big?

As a customer  
I want to find an ATM  
So that I can make deposits or with-drawls outside of banking hours

## Acceptance Criteria:

1. Stop payment on check
2. Find a branch
3. Find an ATM
4. Order new checkbook
5. Get statement < 2 years old

As a customer  
I want to find a nearby branch  
So that I can conduct business in person

As a customer  
I want to stop payment on check so that I can prevent a payment made in error

As a customer  
I want to see my canceled checks online  
So that I can confirm transactions

# User Stories: Invest - Testable

- You need clarity on the story specific done criteria
- Solutions include adding acceptance criteria or better defining the story



?????

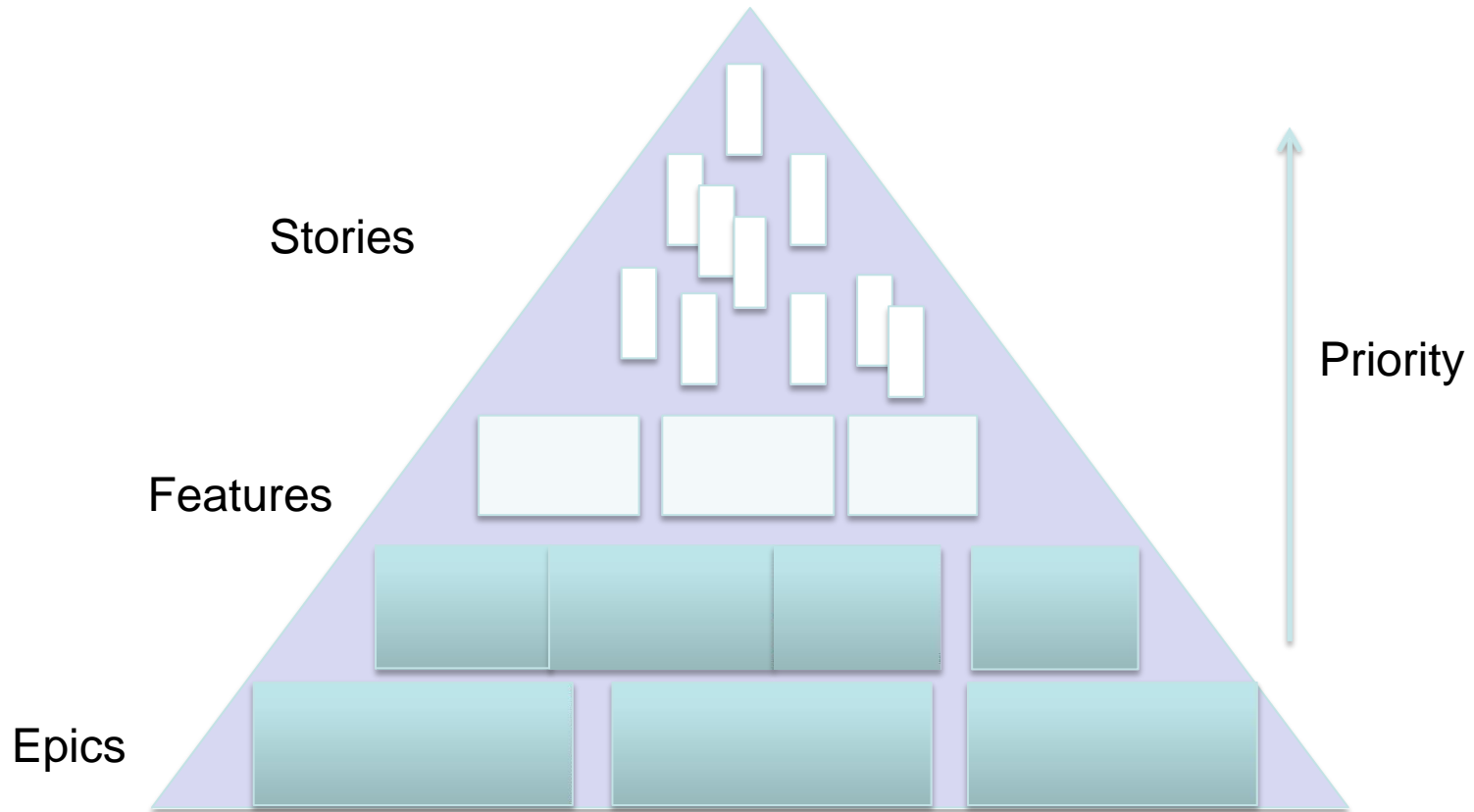
As a registered user  
I want a better looking  
homepage  
So that I don't have to look at  
something so ugly

Have to manually test, but it is  
clear.

## Acceptance Criteria:

1. All text is dark color on light background (no more red on black)
2. Only two different fonts used (instead of seven)

# Epics, Features, Stories



Product backlog

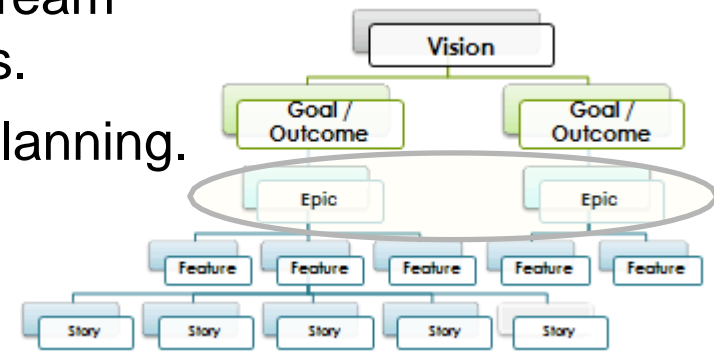
# Epics

**Epics** are high-level features or activities that span Sprints, or even Releases.

- Add a Customer Center for self service.
- Improve database response time by 50%.

## Logistics

- The PO works with stakeholders and the Team to create epics that address desired Goals.
- Epics are often defined prior to Release Planning.
- Often months of effort.



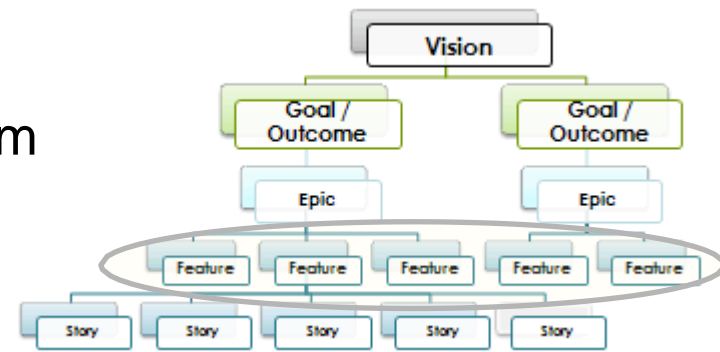
# Features

**Features** are tangible expressions of functionality, but still too large to build.

- As a Bank Customer, find a branch so that I can deposit checks.
- As a Shopper, set up a mobile wallet so I can pay for purchases via Near Field Communications.

## Logistics

- Created by the PO with input from the team
- Often defined prior to Release Planning
- Decomposed over time to smaller Stories
- Typically weeks of effort



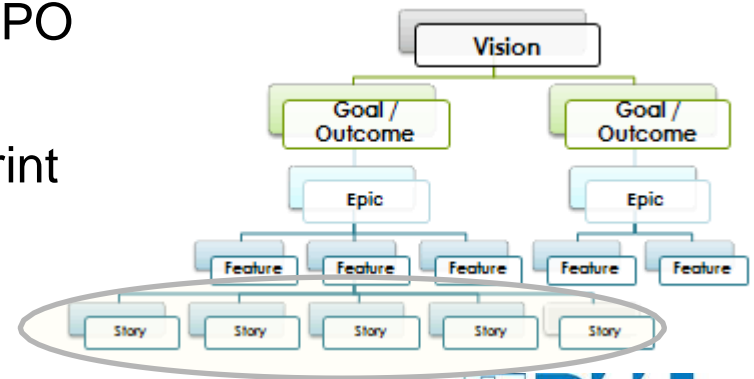
# User Stories

## User Stories are ready for the Team to build.

- As a Bank Customer, find a branch near an address so that I can minimize travel.
- As a Shopper, add an account to my mobile wallet so that I can fund it.

## Logistics

- Refined in backlog grooming sessions by PO and representatives from Team
- Stories should be well-defined prior to Sprint Planning
- Generally about 1-3 days of effort



# Non- Functional Requirements (NFRs)


- **System-wide** nonfunctional requirements may become part of the **Definition of Done**.
- Articulated as tests
- Serve as design constraints

Search response time will not exceed 10 seconds.  
All stories will meet Section 508 accessibility guidelines.

- **Story-specific** NFRs are expressed as **Acceptance Criteria**.

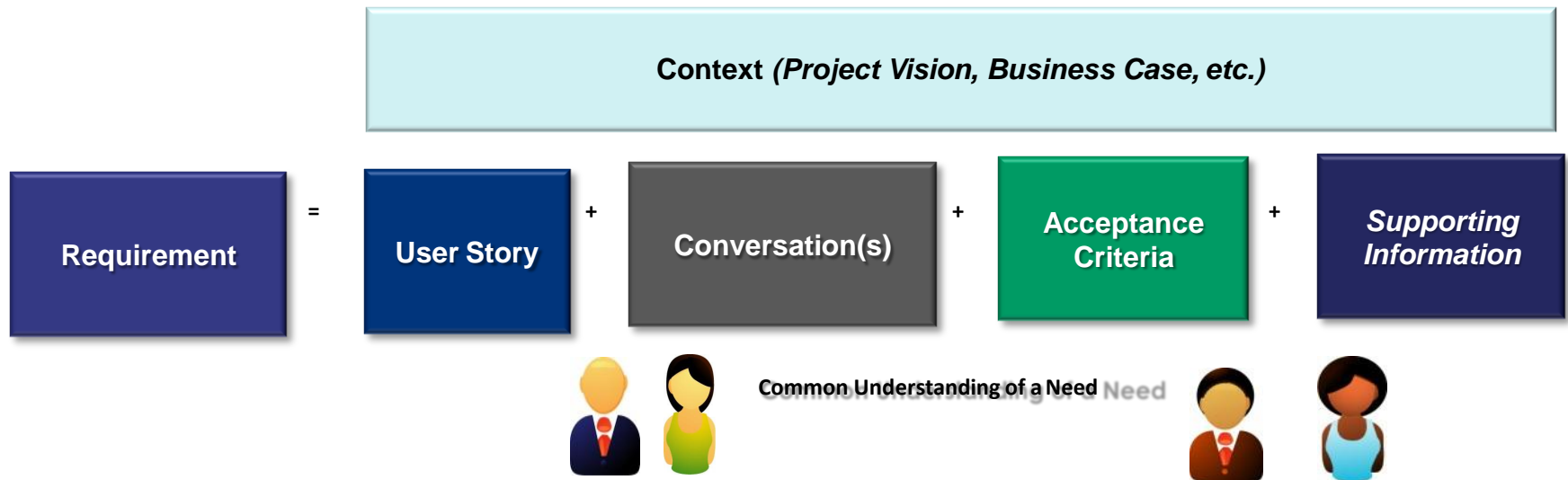


# What User Stories are NOT

User Stories  Requirements

(User Stories  Requirements)

# Requirements, More than Just a Story



# Which Story is Better?

A	B
As a prospect I want to enter my billing information	As a prospect I want to register So I can make purchases online
As a driver I want to find the store with the shortest drive time So I can get there quickly	As a driver I want to find directions to a store on Google Maps So I can get there quickly
As a repeat customer I want to access old orders So that I can rapidly purchase the same products again	As a user I want to have my previous orders stored in the database So they will be there permanently
As a color blind user I want dark text & light background So that I can easily read the text	As a user I want a nice looking site So my aesthetics are satisfied

# Exercise – Write User Stories

Using the Epics we provided for our awesome restaurant finder app:



- Create at least **five** Sprint-sized User Stories based on these Epics.
- Use the “**As a, I can, so that**” format for the User Stories.

---

# Release Planning/Roadmapping

## Story Maps

---

# Mapping Releases with Story Maps

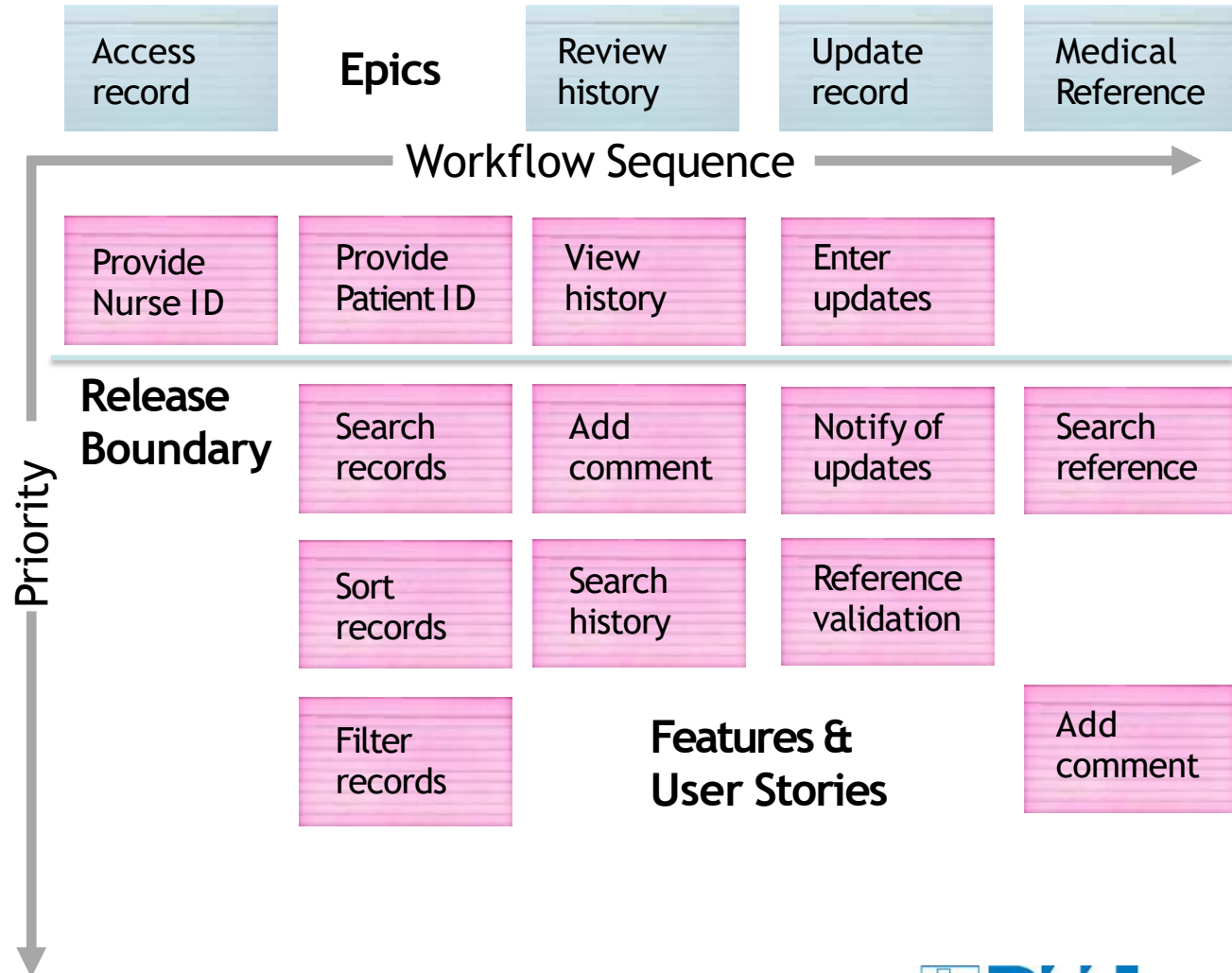
## User Goals

- Minimize the time needed to access patient records
- Minimize the customer inputs necessary to access patient records

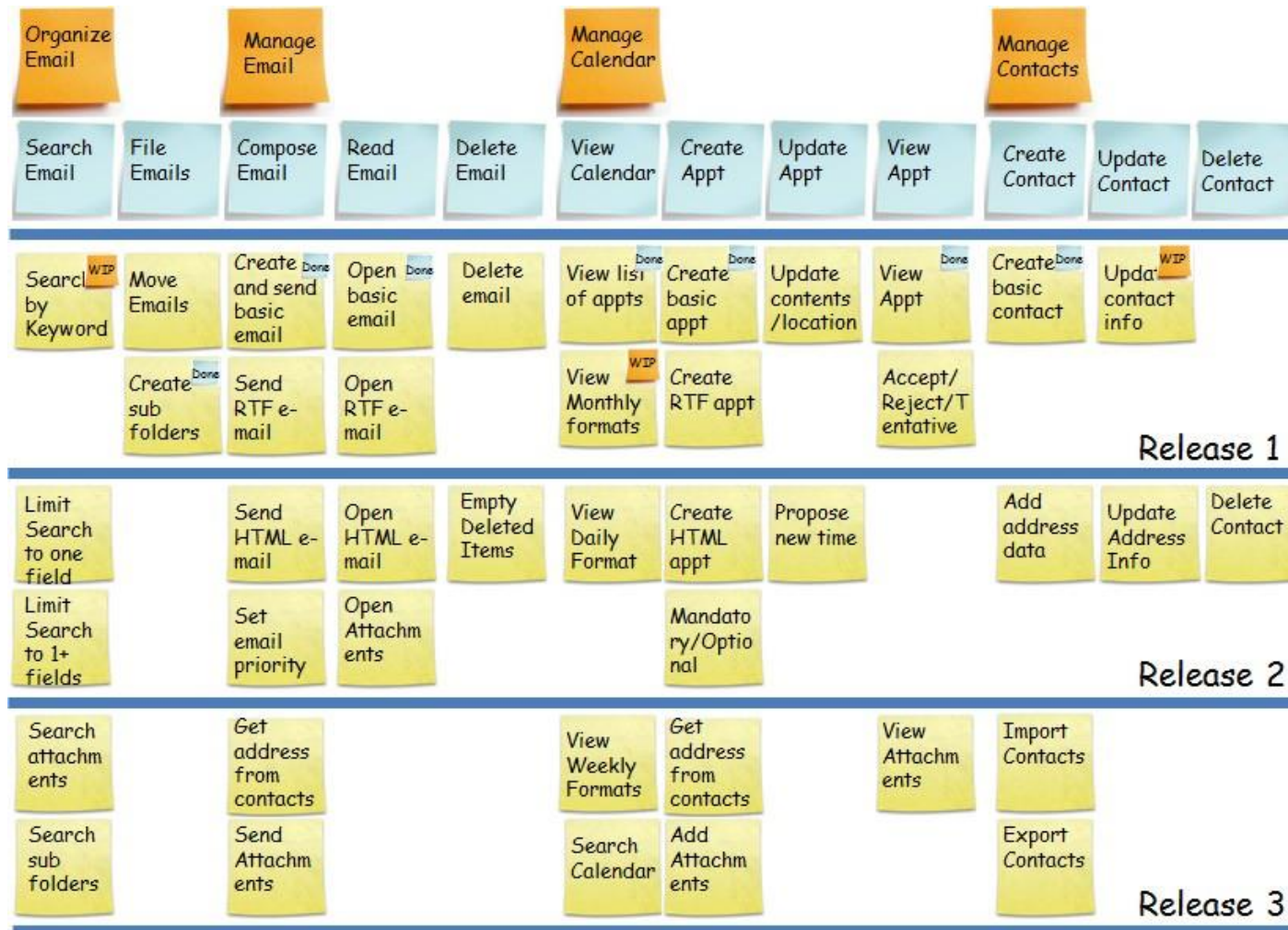
## Persona

Night Nurse  
**Robin**

Robin leaves for work at 6pm, after sleeping during the day. She works a 7pm-7am shift in Labor & Delivery, caring for prospective mothers and their babies. Complex computer apps make Robin grumpy.



# Another Example



Thanks to Winnipeg Agilist for this image

# Prioritization Parameters

**Value** is influenced by many things:

- **Time sensitivity** – Build features that decay in value over time earlier.
- **Uncertainty & Risk** – Use “spikes” to test market or technical viability for critical, risky features.
- **Size** – All else being equal, do the shortest first.
- **External Dependency** – Third party or support group dependencies can be immutable constraints.



# Exercise – Story Mapping

## Plan your **first few Releases** with a **Story Map**.

- Place **Epics** or **High Level Activities** at top, in order of their natural workflow as appropriate.
- Place **User Stories** underneath the Epics that they support, from top to bottom by value.
- **Group the stories** for your first Release (MVP) and subsequent Releases, describing the targeted benefits of each at a Roadmap level.

---

# Product Backlog Management

Refining, Accepting & Testing

---

# Product Backlog Essentials

## Adding User Stories

- Anyone can suggest backlog items
- Product Owner prioritizes them

## Estimating & Elaborating

- High-priority items are estimated and described most precisely, since they will be worked on first
- Low-priority items are generally estimated and described coarsely

## Prioritizing

- Ordering is driven mainly by business value and risk reduction.

**High priority** items  
are better defined



#	Backlog Item	Estimate
1	Create login screen	1
	...	
20	Allow user to browse recently viewed items	8
	...	
60	Add personalization	30 (or so)



**Low priority** items  
are often “epics”

# Progressive Elaboration

3-4 Sprints ahead or more

## Ideation

Market Trends  
Prototypes  
Focus Groups  
User Experience  
Basic Workflows  
Vision  
Business Outcomes  
Release Timing and Goals  
Product Architecture  
Epics and Features

Marketing/Sales,  
Product Management,  
Product Owners,  
Architects

~1-2 Sprints ahead

## Maturation

User Story Decomposition  
User Story Maturation  
Acceptance Criteria  
Test Cases  
Dependencies  
Story Mapping  
Prioritization  
Epic Estimation  
Backlog Development

Product Owners,  
Architects, Dev Leads,  
QA Leads, UX/Analysts

Current Sprint

## Execution

Sprint Planning  
Sprint Estimation  
Daily Standups  
Software Development  
Testing  
Burndowns  
Documentation  
Product Demos  
Retrospectives

Leads, UX/Analysts,  
Dev Team Members

# Maturation of a User Story

## User Stories

Created during upfront and ongoing Discovery

As a user  
I want to create an account  
So that I can shop online

*Priority – 1 (High)*

*Estimate - 5 Points*

## Acceptance Criteria

Sprint Pre-Planning (Backlog Grooming)

- Phone # in US format, contains no alpha characters, minimum 10 digits
- First name, Last name and email address required
- Email specified in standard format

## Testable Examples (ATDD)

Sprint Pre-Planning (Backlog Grooming)

Name	Phone	Email	Valid
John Smith	215-555-1212	<a href="mailto:jsmith@ls.com">jsmith@ls.com</a>	TRUE
Smith	215-555-1212	<a href="mailto:jsmith@ls.com">jsmith@ls.com</a>	FALSE
John	215-555-1212	<a href="mailto:jsmith@ls.com">jsmith@ls.com</a>	FALSE
	215-555-1212	<a href="mailto:jsmith@ls.com">jsmith@ls.com</a>	FALSE
John Smith		<a href="mailto:jsmith@ls.com">jsmith@ls.com</a>	FALSE
Smith		<a href="mailto:jsmith@ls.com">jsmith@ls.com</a>	FALSE

## Sprint Tasks

Sprint Planning

- Design UI Mock Up
- Write online help text
- Develop CSS/HTML
- Develop validation criteria
- Create database tables
- Code test fixtures
- Code & Test

# Sample Story Maturation Look-A-Head

The tactical act of getting a story ready is often performed as a **two sprint look-a-head** by an amigos team

Sprint n

Select User Story 999 for Sprint n + 2  
Re-estimate it, sharpen story & acceptance criteria

The PO and 3+ Amigos look-a-head and select story 999 for inclusion for Sprint n + 2. They do cleanup on the story.

Sprint n + 1

Create testable example and other supporting material for 999  
Get sign off from external parties

The 3+ Amigos further support the story and the PO gets appropriate sign offs.

Sprint n + 2

Develop  
User Story 999

Story 999 makes it's way into a sprint and it is built.

# Approaches to Splitting Stories

- 1.To Demonstrate Progress:** UX with no validation or save
- 2.CRUD:** Create, report, update & delete (e.g. split of manage)
- 3.Basic to Advanced:** Sort by one field (name), sort by any one field (name, date, etc.), sort by a combination of fields
- 4.Use Case scenarios:** Happy path, alternates, exceptions
- 5.Workflow steps:** Find book, see details, purchase
- 6.Importance:** Credit card, split across cards, automatic billing
- 6.UI complexity:** Manual coordinates, interactive web map
- 7.Spike and Build:** Research credit card processing, implement

# Split some Stories

1	As a ... I want to manage my widgets	
2	As a report viewer I want to filter my report by any combination of columns	
3	As a possible room renter I want to find and book a room	
4	As a customer I want to pay electronically	



# Split some Stories

5	As a low budget vacation traveler I want to find flights using a range of dates	
6	As a credit card purchaser I want to pay by Amex, MasterCard, Visa or Discover	
7	As a frequent user I want to personalize my experience	

# Getting to Ready

3+ Amigos (BA, QA, Dev) can serve as a readiness team to look ahead and ensures we are:

1. Ready to prioritize
2. Ready to estimate / right size
3. Ready to build



# When is a Story “Ready”?

## Definition of “Ready”

- Choose the **few items** that your team finds most useful in Sprint Planning.
- Confident and **quick Sprint Planning** and **smooth Sprints** that produce polished results are your goals.

☒ ☒ Interaction Diagrams

☐ ☐ Prototypes

☐ ☐ Wireframes

☒ ☒ Sample Data

☒ ☒ Testable Examples

☒ ☒ Acceptance Criteria

☐ ☐ State Diagrams

☒ ☒ Small Enough

☐ ☐ Agreement from other teams

☐ ☐ Approvals (Compliance, Security, Brand Mgmt, etc.)

☐ ☐ Dependency List

☐ ☐ Stakeholder signoff

# When is a Story “Done”?

## Definition of “Done”

- A **shared definition** and compact between Teams and Stakeholders
- Denotes what stories require to be **accepted**
- Ideally represents “**potentially releasable**” or even released state

☒ Acceptance Criteria are met

☒ Cleared by QA

☒ Accepted by Tactical PO

☒ Accepted by Strategic PO

☐ Live for A/B Testing

☐ Final Deployment

☐ Training Script

☒ Pair reviewed

☐ Peer Reviewed

☒ Integrated

☐ Lightweight usability tested

☒ Automated testing in place

☒ User documentation created

☒ Ops documentation created

# “Done Done” at Release Level

## What are the minimum criteria for each Release?

- Testing/quality targets
- Performance targets
- Operational (e.g. sales/marketing) deployment goals
- Required documentation & artifacts
- Regulatory compliance targets

### Example Release Criteria

- System response on all level 1 functions within 5 seconds
- No Severity 1-3 bugs in Firefox 2+, Chrome, IE 7+ or Safari 3+
- No Severity 1 or 2 bugs found during final month
- Full compliance with accessibility guidelines in Section 508

---

# Acceptance Criteria & Testing

---

# Acceptance Criteria Example

As a user

I can cancel a registration

So that I don't have  
to pay

- ☐ Premium member can cancel the same day without a fee
- ☐ Non-premium member is charged 50% of first day for a same-day cancellation
- ☐ Email confirmation is sent to members primary and secondary email addresses
- ☐ Hotel is notified of any cancellation

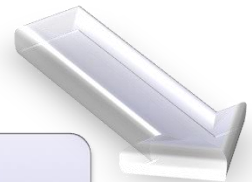
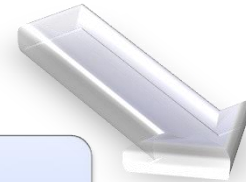
# Modern Agile Acceptance Model

**Conditions of Satisfaction –**  
Broad Terms

**Acceptance Criteria –**  
Further Refined

**Examples –**  
Actual scenarios or data

**Executable Examples –**  
Ready to automate

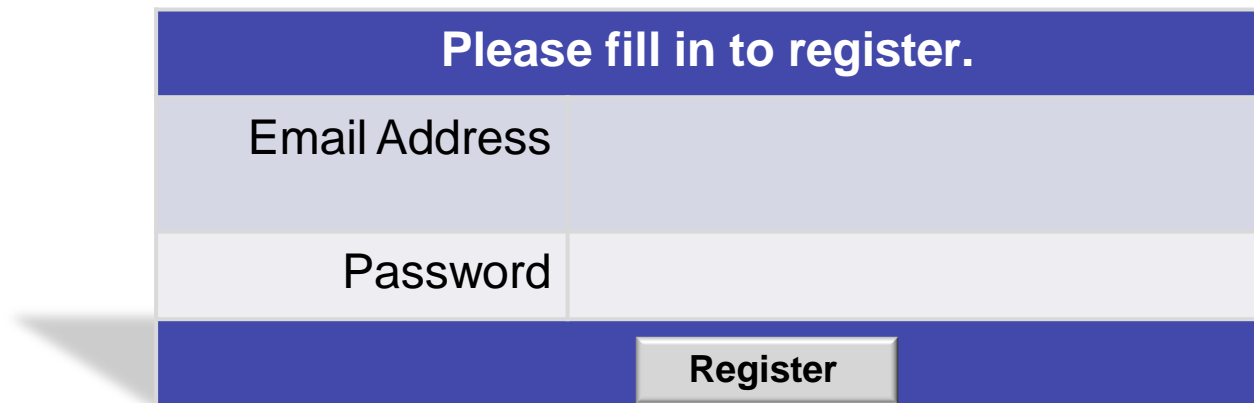




# Acceptance – Conditions of Satisfaction

## Testing a Registration Function

- What constraints should we impose?
- Business stakeholders and PO agree that **passwords should not be easy to crack.**



Please fill in to register.

Email Address	
Password	
<input type="button" value="Register"/>	

# Acceptance – Acceptance Criteria

## Acceptance Criteria, the Details

The PO works with testers and developers from the team, business stakeholders, users or SMEs to come up with this definition of “**not easy to crack**”:

- **Must be at least 8 characters and no more than 12**
- **Must contain only alphanumerics and the period**
- **Must contain at least one digit**
- **Must contain at least one alpha character**

Please fill in to register.	
Email Address	
Password	????????????

# Acceptance – Examples Written

- Examples, Making it *Real*
- Actual examples are the ultimate way to communicate requirements

Password	Expected	Expected Message	Comment
abc123	Invalid	You password must be at least 8 characters long, and no more than 12 characters long.	
abcdefghi	Invalid	Your password must contain at least one character and one number.	
1aaaaaaaaa	Valid		Why valid?
ajx972dab	Valid		

- Team Members with testing and development expertise take the lead on creating examples, with the Product Owner verifying
- Based on these examples, Acceptance Criteria can be refined
- These criteria act as good starting test cases too

# Acceptance – Testable Examples

- Executable Example, Making it *Repeatable*
- Examples that can be executed are the final step

Given the “Unregistered User” user has navigated to the “register” page

When entering “newuser” in the “Username” field

And entering “abc123” in the “Password” field

And entering “abc123” in the “Confirm Password” field

And pressing the “Register” button

Then the text “Thank you for Registering” should appear on page

And the URL should end with “use/accountPage”

# Acceptance Test Driven Development (ATDD)?

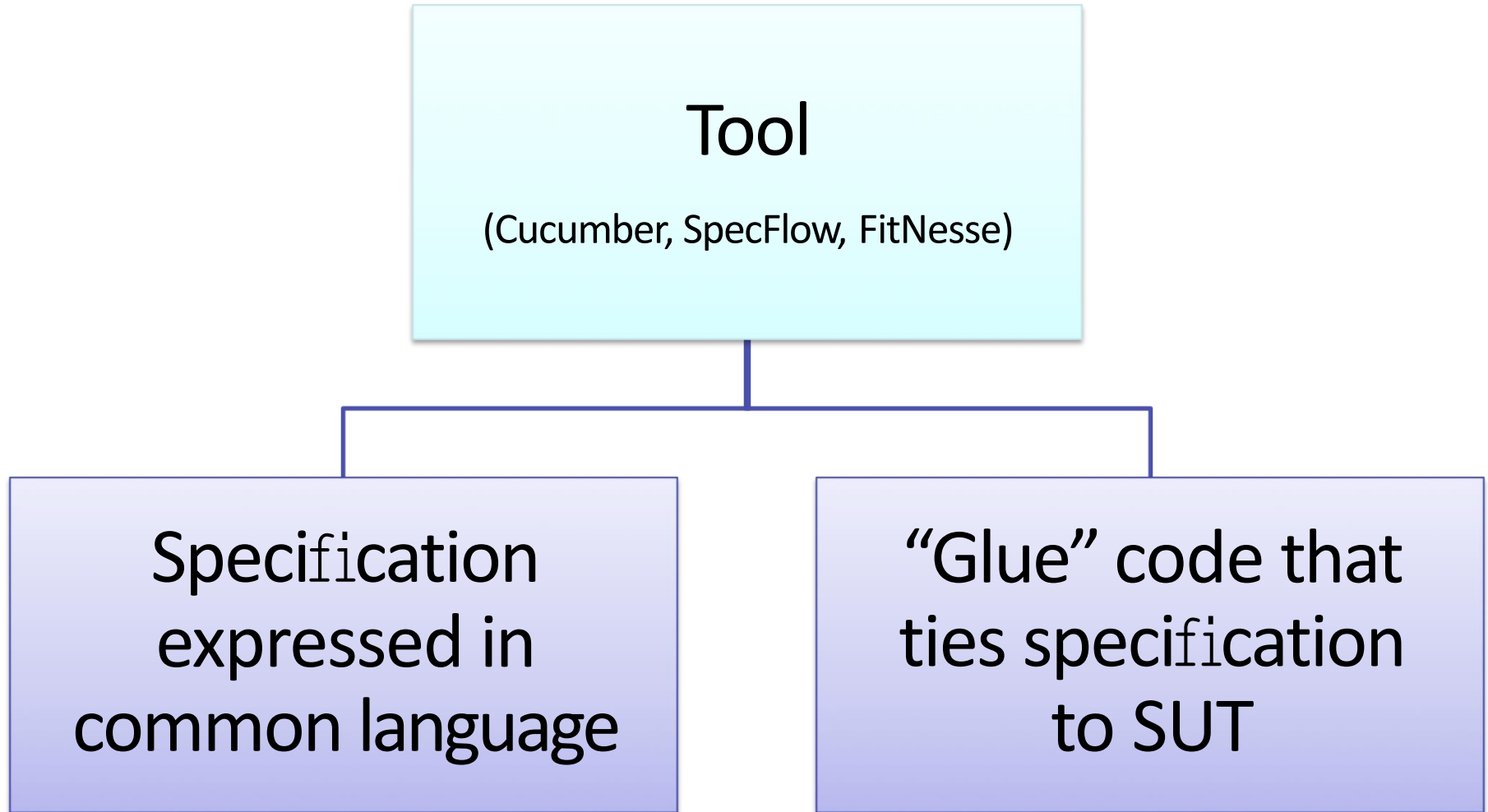
A practice in which the whole team **collaboratively** discusses acceptance criteria, with examples, and then distills them into a set of concrete acceptance tests **before** development begins.

- Elisabeth Hendrickson

# ATDD - Key Characteristics

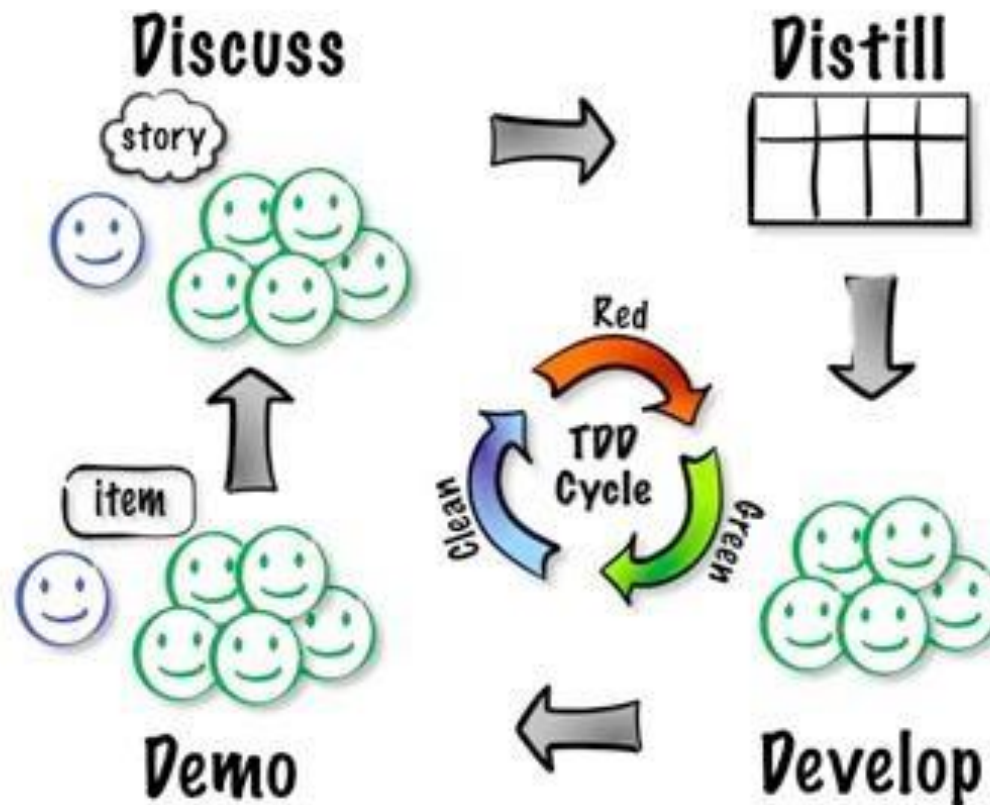
1. Elicit details from the business stakeholder(s) about their expectations
2. Distill acceptance criteria into automatable tests expressed in a natural language
3. Wire the tests to SUT with “glue” code (as part of implementation)

# Automated Acceptance Tests



# ATDD Cycle

Acceptance Test Driven  
Development (ATDD) Cycle



ATDD cycle model developed by James Shore with changes suggested by Grigori Melnick, Brian Marick, and Elisabeth Hendrickson.



# Exercise – Write acceptance tests

## Create **testable specifications**

Using the **User Stories** you defined

- **Write acceptance criteria** for each story
- **Create at least two scenarios** to act as examples and tests, using this format:

### **Scenario: Recent Account Activity**

**Given** I am a registered user “Jsmith”

And I am logged in with password “xyx123”

And I have had account activity in the last 45 days

**When** I click the “Recent Activity” button

**Then** I should see the “Account Activity” Page

And I should see a list of my activity over the last 45 days

# Concluding Thoughts

## Common pitfalls with writing user stories

- Forgetting about the User
- Too much Detail
- Lack of Conversation
- No Acceptance Criteria



Thank you for attending this session.  
We hope you found this presentation  
added value to your knowledge of  
Project Management.

- Take a few moments to complete the Session Survey. We appreciate and value your feedback.
- Hand in your completed survey to **Registration**, you will receive a free raffle ticket for one of the drawings to be held in the Vendor Expo (see Conference Program Guide for details).