As a (USER)
I want to (DO THIS)
so that I can (ACHIEVE THAT)

User Stories

Short, simple descriptions of a feature told from the perspective of the customer

PEGA®

What is a User Story?

From Mike Cohn:

- “User Stories are a part of an Agile approach that helps shift the focus from writing about requirements to talking about them…”
- “All Agile User Stories include a written sentence or two, and, more importantly, a series of conversations about the desired functionality.”

Source: https://www.mountaingoatsoftware.com/agile/user-stories

The 3 C's of User Stories

- CARD
- CONVERSATION
- CONFIRMATION

The INVEST Model

INVEST is an acronym that captures the ideal qualities of User Stories:

- **Independent:** The User Story is not dependent on other Stories.
- **Negotiable:** The User Story can be changed, rewritten, or split (prior to being committed to a sprint).
- **Valuable:** The User Story must deliver value to the end user.
- **Estimable:** The Development Team must be able to estimate the User Story’s size.
- **Small:** Every User Story has to be sized appropriately to fit into a Sprint.
- **Testable:** The User Story is capable of being tested.

What's the difference between User Stories and Requirements?

	USER STORIES	REQUIREMENTS
Who creates?	"3 Champions" to Full Teams	Typically only the Business
Level of detail	Just enough; open for negotiation	High level of detail, specific; tends to cover all possible combinations
Time to create	Quick, placeholder for further conversations and elaboration	Can be laborious, taking significant time to complete
When to create?	Create when/as needed; reflect current needs	Created far in advance; reflect needs at a particular point in time, typically in the past
Primary Info Transmission	Collaboration emphasized, write down only what's needed after discussion	Written documentation
Hand-offs	Low amount of hand-off; expectation for collaboration	Tendency for high amount of hand-offs – business to IT, IT to Quality, Quality to…
Change Control	Experiment, learn, expectation for change	Lock down then implement significant effort process for changes

User Story Format

- As a…
  [Who? - Insert the User]

- I would like to…
  [What? - Insert what they want to do]

- so that…
  [Why? – Insert the Business Value]

Example Stories – Hypothetical Promotion Enrollment:

As a...
Premium Member

I would like...
to be automatically enrolled in all eligible promotional offers

so that...
I can take advantage of benefits without having to manually enroll.

As a...
Non-premium Member

I would like...
to enroll in promotional offers

so that...
I can take advantage of special incentives on my purchases

Implementation Details vs. Value of Implementation

As a...
Supervisor

I would like...
to run a report

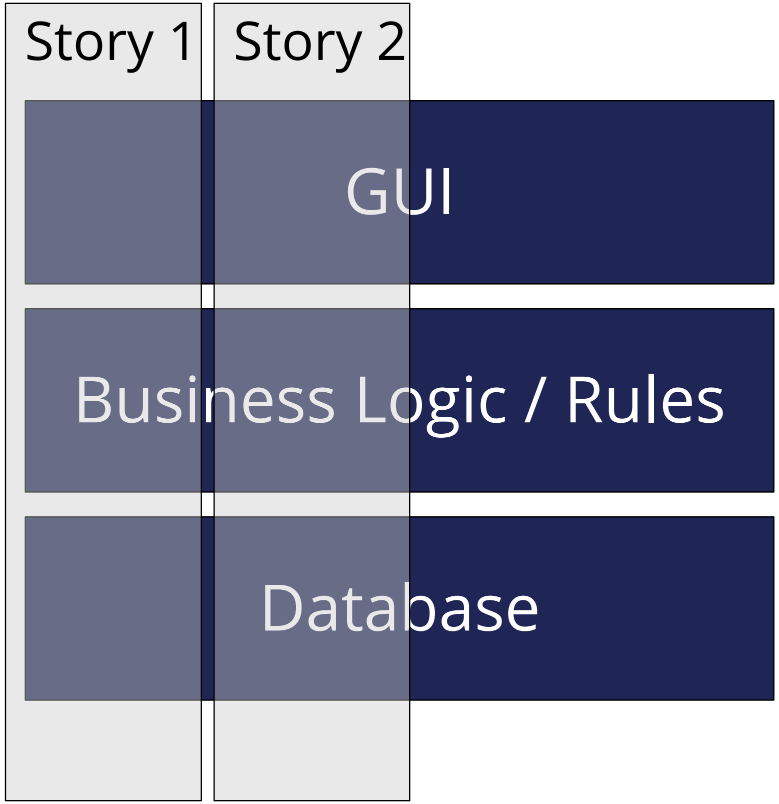
so that...
I can see how many registrations for promotions have been made for Premium and Non-Premium members.

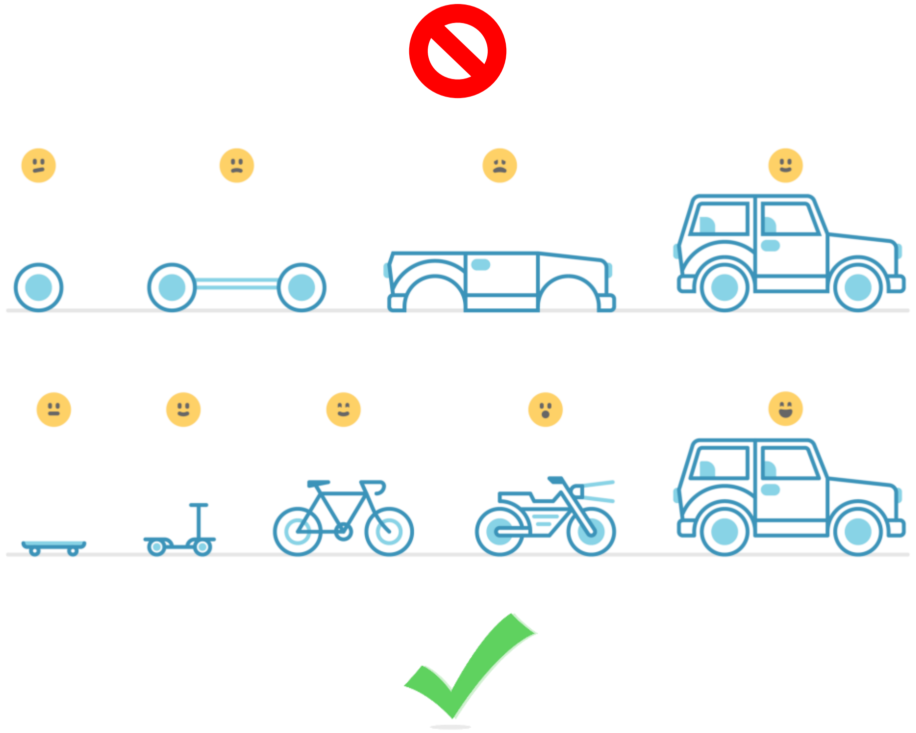
As a...
Call Center Supervisor

I would like...
to see in real-time how many customers have registered for a promotion

so that...
I can make adjustments as needed to promotion enrollment scripts and awareness.

Deliver User Stories as Vertical Slices of Functional Product:





User Story descriptions are not enough…

You also need Acceptance Criteria to better describe expected outcomes.

- **Acceptance Criteria characteristics:**
  - A list of outcomes that enable a Product Owner to accept a User Story
  - Adds clarity to the story’s deliverables
  - Provides a guide to developers for effective testing
  - Helpful for further documentation
  - A good tool for splitting up work and negotiating the schedule of deliverables
- **Suggested format:**
  - “This story is done when…[insert outcomes, not implementation details]”

Example Acceptance Criteria – Hypothetical Promotion Enrollment:

As a…
Premium Member

I would like…
to be automatically enrolled in all eligible promotional offers

so that…
I can take advantage of benefits without having to manually enroll.

This story is done when:

- Premium member is enrolled automatically 1 week before the promotion start date.
- Premium member receives email they've been enrolled.
- Account rep for top 10% of Premium members is notified.

User Stories are not "requirements" nor detailed specs…

- **They are intended to be "placeholders for further conversations" – they are not intended to be fully defined specifications down to detailed minutiae.**
  - Details of exactly what will be required will be built in collaboration between business and IT.
  - Focus on the "why" (outcomes) and not the "how" (implementation details).
- **They will be progressively elaborated over time:**
  - User Stories at the top of the product backlog (highest priority) will have the most detail.
  - User Stories at the bottom of the product backlog (lowest priority) may have just a title.
- **How much detail is needed for a User Story to be ready?:**
  - Enough for the team to Size and Task the story.
  - This can be in conflict with traditional DCO practices.

Special Story Type – Spikes

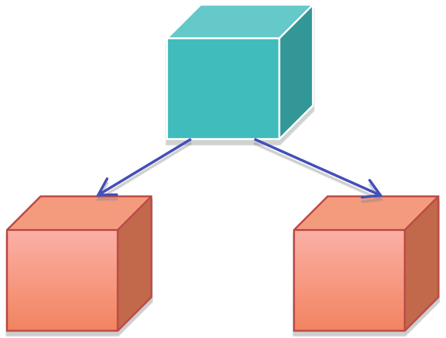
- **There are times when you may not know enough to deliver a User Story (incremental value) directly to a customer.**
- **You may want to gain clarity on…**
  - Do we have a problem?
  - Does our proposed solution solve that problem?
  - Is our proposed solution technically feasible?
- **Use a Spike!**
  - Be sure that the "so that…" clause clearly identifies what you are going to do with the information once you have it.
  - Acceptance Criteria is still necessary.
    - What is it that you need to know?
- **This is a special case – use Spikes sparingly**
  - The presence of multiple spikes indicates you may not have enough clarity on the problem and/or solution

User Stories must fit within the Sprint

The Goal is to deliver potentially shippable product at the end of each iteration…

Complex Problems

“We just don’t know enough…”

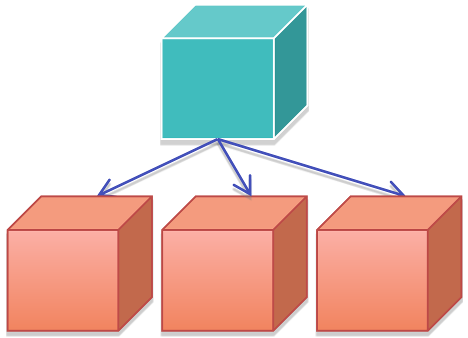


Research Spike
(do first)

Functional Story(ies)
(do after spike)

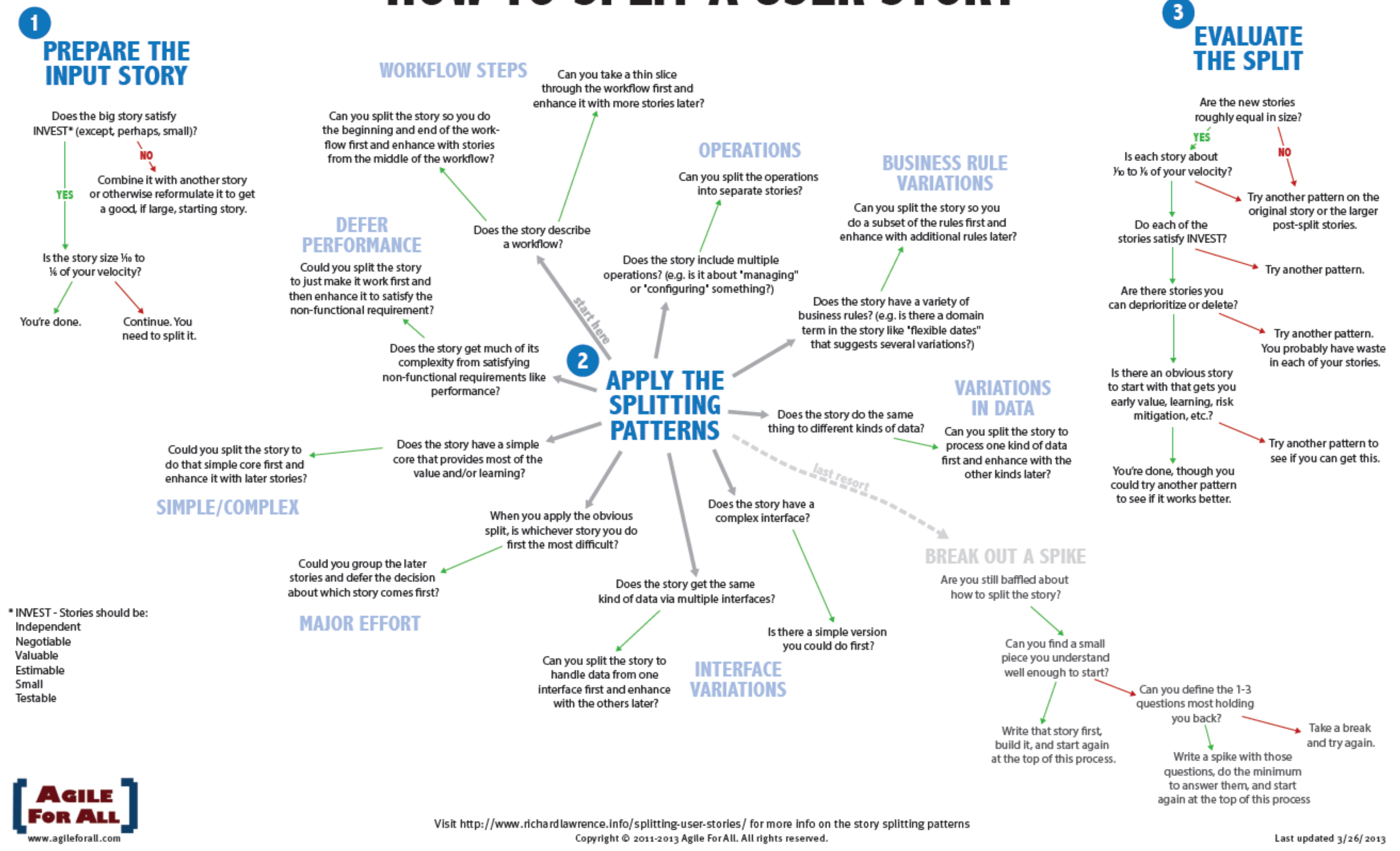
Compound Problems

“Its just too big.”



Multiple Functional Story(ies)
(prioritized)

HOW TO SPLIT A USER STORY



1 PREPARE THE INPUT STORY

Does the big story satisfy INVEST* (except, perhaps, small)?

NO → Combine it with another story or otherwise reformulate it to get a good, if large, starting story.

YES → Is the story size 1/10 to 1/6 of your velocity?

→ You're done.

→ Continue. You need to split it.

2 APPLY THE SPLITTING PATTERNS

start here

WORKFLOW STEPS

Does the story describe a workflow?

→ Can you split the story so you do the beginning and end of the workflow first and enhance with stories from the middle of the workflow?

→ Can you take a thin slice through the workflow first and enhance it with more stories later?

OPERATIONS

Does the story include multiple operations? (e.g. is it about "managing" or "configuring" something?)

→ Can you split the operations into separate stories?

BUSINESS RULE VARIATIONS

Does the story have a variety of business rules? (e.g. is there a domain term in the story like "flexible dates" that suggests several variations?)

→ Can you split the story so you do a subset of the rules first and enhance with additional rules later?

VARIATIONS IN DATA

Does the story do the same thing to different kinds of data?

→ Can you split the story to process one kind of data first and enhance with the other kinds later?

INTERFACE VARIATIONS

Does the story have a complex interface?

→ Is there a simple version you could do first?

Does the story get the same kind of data via multiple interfaces?

→ Can you split the story to handle data from one interface first and enhance with the others later?

MAJOR EFFORT

When you apply the obvious split, is whichever story you do first the most difficult?

→ Could you group the later stories and defer the decision about which story comes first?

SIMPLE/COMPLEX

Does the story have a simple core that provides most of the value and/or learning?

→ Could you split the story to do that simple core first and enhance it with later stories?

DEFER PERFORMANCE

Does the story get much of its complexity from satisfying non-functional requirements like performance?

→ Could you split the story to just make it work first and then enhance it to satisfy the non-functional requirement?

BREAK OUT A SPIKE

last resort

Are you still baffled about how to split the story?

→ Can you find a small piece you understand well enough to start?

→ Write that story first, build it, and start again at the top of this process.

→ Can you define the 1-3 questions most holding you back?

→ Write a spike with those questions, do the minimum to answer them, and start again at the top of this process

→ Take a break and try again.

3 EVALUATE THE SPLIT

Are the new stories roughly equal in size?

NO → Try another pattern on the original story or the larger post-split stories.

YES → Is each story about 1/10 to 1/6 of your velocity?

→ Try another pattern.

→ Do each of the stories satisfy INVEST?

→ Try another pattern.

→ Are there stories you can deprioritize or delete?

→ Try another pattern. You probably have waste in each of your stories.

→ Is there an obvious story to start with that gets you early value, learning, risk mitigation, etc.?

→ Try another pattern to see if you can get this.

→ You're done, though you could try another pattern to see if it works better.

* INVEST - Stories should be:
- Independent
- Negotiable
- Valuable
- Estimable
- Small
- Testable

AGILE FOR ALL www.agileforall.com

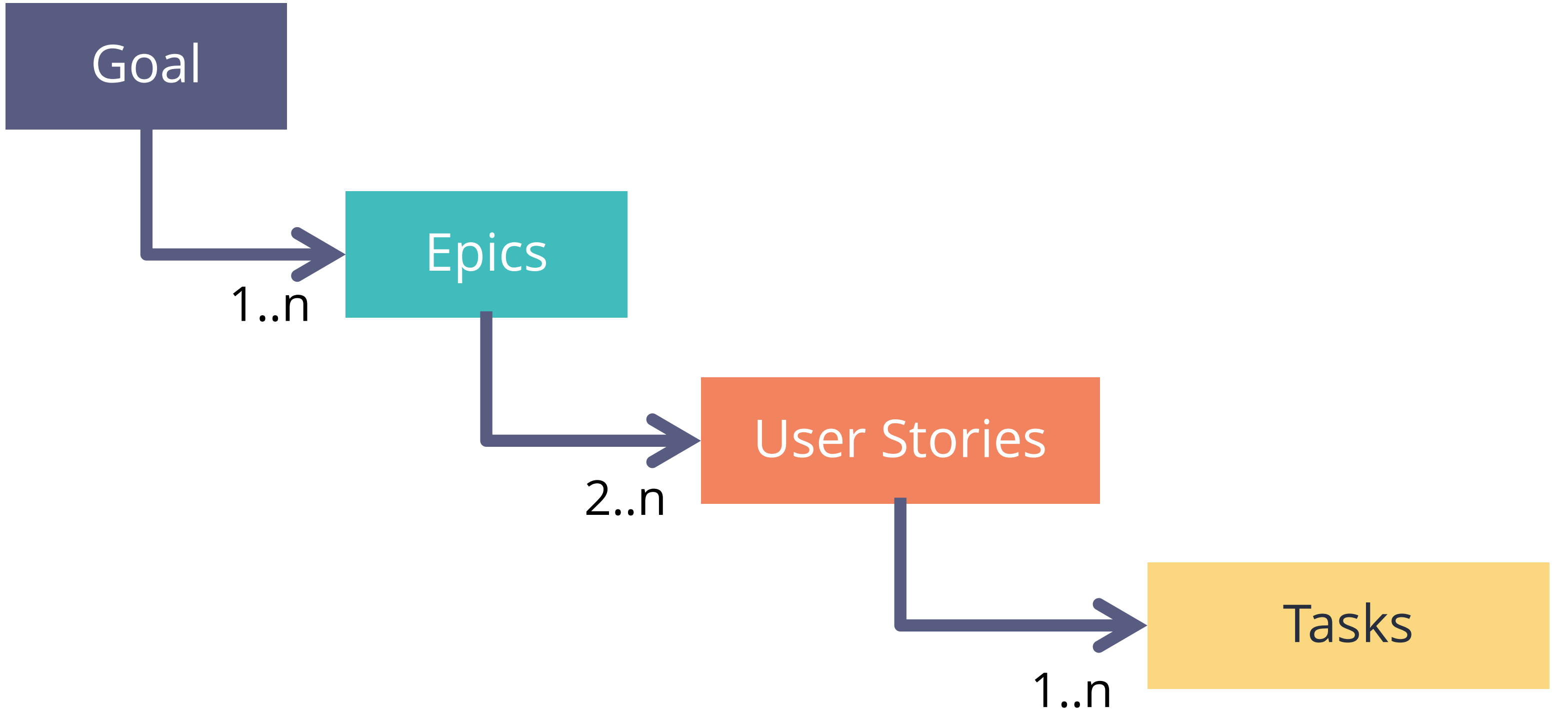
Visit http://www.richardlawrence.info/splitting-user-stories/ for more info on the story splitting patterns

Last updated 3/26/2013

Do you have an Epic?

- **What is an Epic?**
  - Typically represents a large chunk of work
  - Consists of (2) or more User Stories
- **What are the characteristics of an Epic?**
  - Still follows the same format as a User Story (title, description, Acceptance Criteria)
  - Lives in the Product Backlog
  - Unlike User Stories, the delivery of Epics can span multiple sprints
  - Often start with Epics that eventually get decomposed into User Stories
    - But, can also take a group of User Stories and create an Epic as well

Relationships between Epics and User Stories



Definition of Done (DoD)

- **Having User Stories and Acceptance Criteria are usually not enough to say a story is releasable.**
- **Definition of Done:**
  - Set of activities required for User Stories to be considered complete so that the increment of product functionality can be "potentially shipped" if desired
- **Some examples of a Definition of Done (DoD) could be:**
  - Code/Rules reviewed
  - Unit testing added to automation testing suite and all Unit tests pass (not just for code added this sprint)
  - Regression test suite updated
  - User documentation (as required) is completed/updated
- **The Development Team and Product Owner (PO) need to come to agreement on DoD before planning the first sprint**
  - DoD may vary from team to team – that's OK as long as it's transparent
  - If multiple teams/POs are involved, there needs to be common agreement
  - DoD often evolves over time as Dev Teams become more proficient

PEGA®

Build for Change®