

User Stories101

What is a User Story?



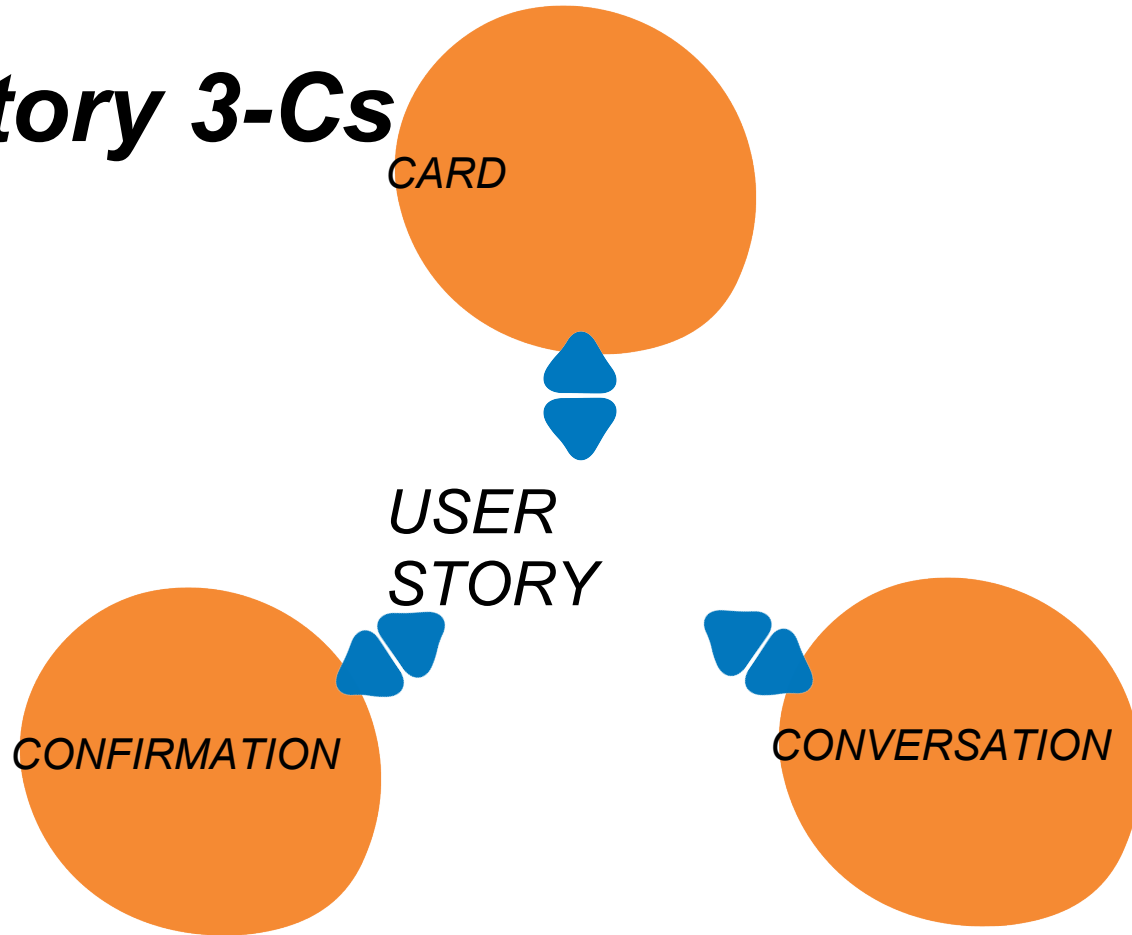
User stories

- A tool for **iterative** development
- Represents a **unit** of work that should be developed
- Helps **track** that piece of functionality's lifecycle
- It is a token for a **conversation**, a placeholder for a **conversation**

Why should I use them?

- It is a piece of customer-visible functionality written in common language
Universally understood
- Ensure you only build things for a reason
Prevents waste
- Simple and flexible
Minimum overhead
- Proven way of gathering requirements on agile projects
Effective teams

Story 3-Cs



Story 3-Cs

CARD

Physical token
Used in planning
Reminder for a conversation
Often annotated

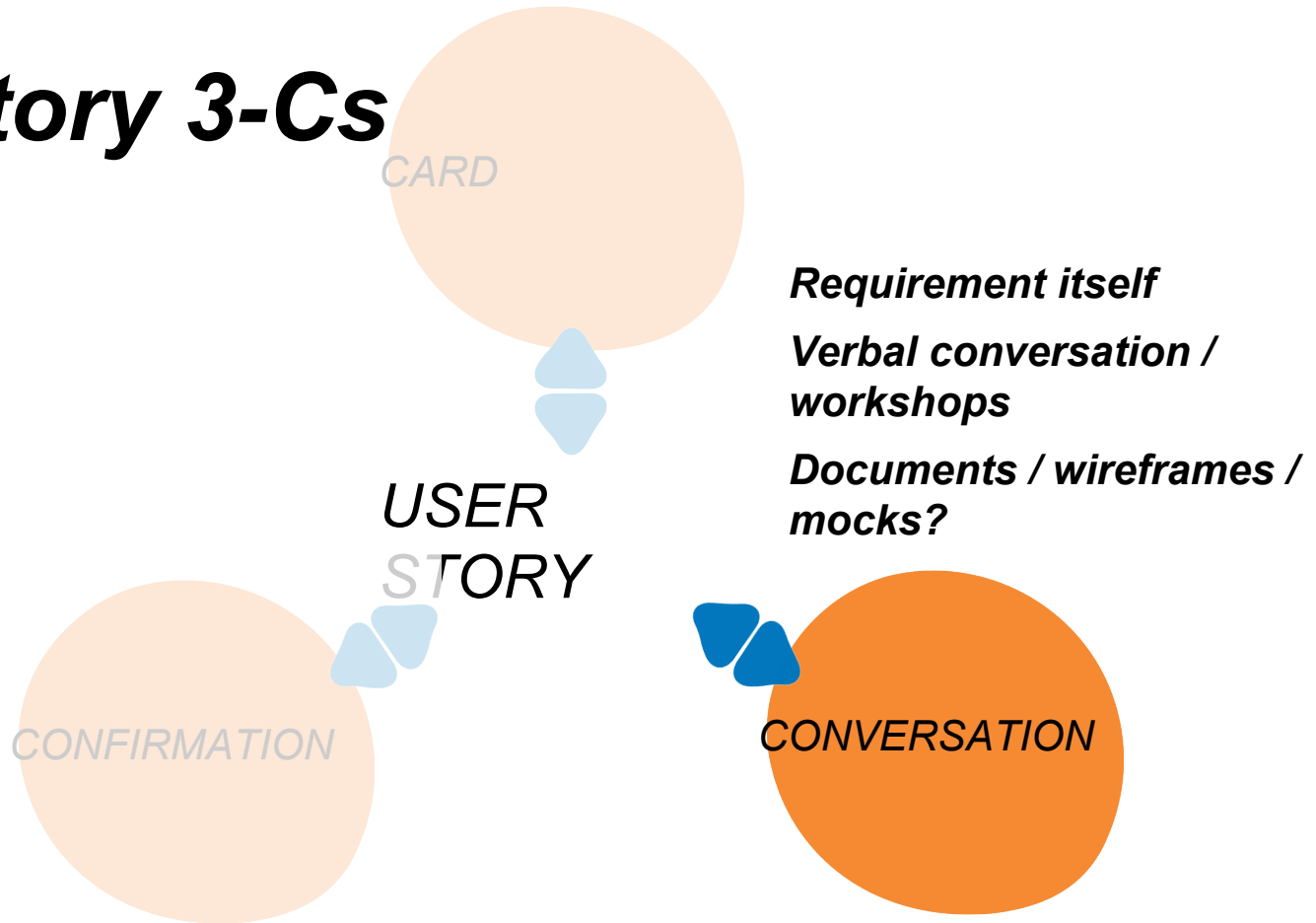


USER
STORY

CONFIRMATION

CONVERSATION

Story 3-Cs



Story 3-Cs

Acceptance criteria
Determine done

CONFIRMATION

USER
STORY

CARD

CONVERSATION

In a nutshell

Moving from statements about what the system should do...

The system shall <do something>

to a concise description of a piece of functionality that will be
valuable to a user (or owner) of the software

*As a <role>,
I want to <business goal>,
so that <value/motivation>*

As a <role>

I want to <business goal>

So that <value>

INVEST Principle

- | | | |
|----------|-------------|--|
| I | Independent | <i>No overlap – order is ok!</i> |
| N | Negotiable | <i>No contract. Details can change.</i> |
| V | Valuable | <i>Incremental benefit to something.</i> |
| E | Estimable | <i>Relative size to other stories.</i> |
| S | Small | <i>Shouldn't be bigger than an iteration.</i> |
| T | Testable | <i>Should be able to tell when it is done.</i> |

INVEST Principle



Valuable

*Someone is benefiting from
what we are building*

*Why are we building
it?*

ThoughtWorks®

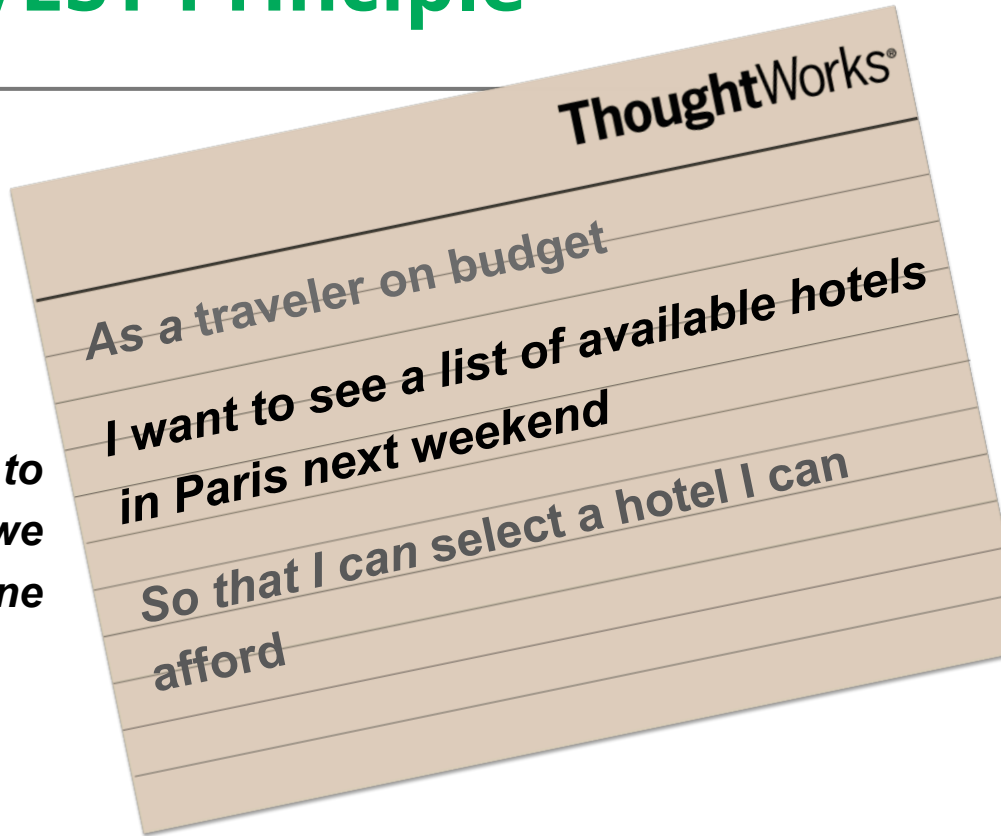
*As a traveler on budget
So that I can select a hotel I can
afford*

INVEST Principle



Testable

*Now we know what to
do and know when we
will be done*



INVEST Principle



Small

*Just enough to get
feedback and avoid
waste*



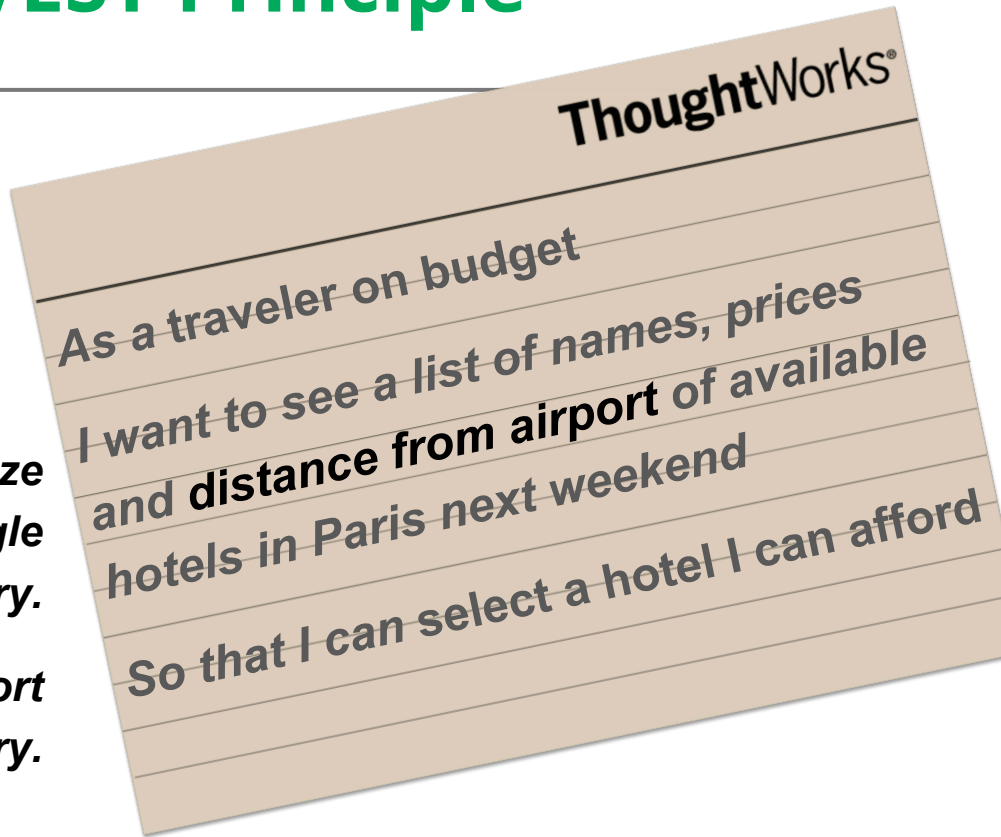
INVEST Principle



Independent

*Should be able to realize
the value of each single
story.*

*Distance from the airport
can be in another story.*



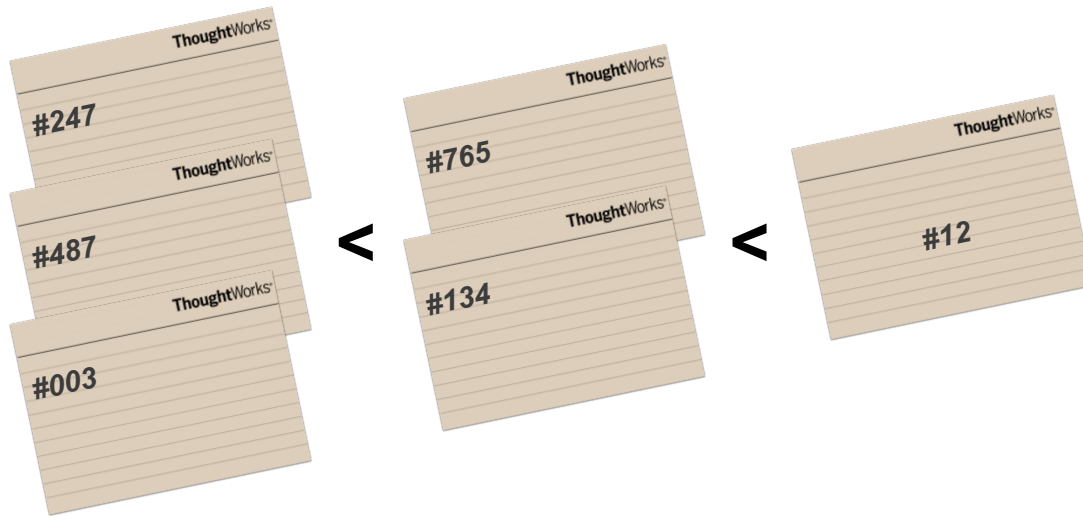
INVEST Principle

N Negotiable



INVEST Principle

E Estimable



Getting practical

I Independent

N Negotiable

V Valuable

E Estimable

S Small

T Testable



V Valuable

T Testable

S Small

I Independent

N Negotiable

E Estimable

What is in a user story

- “As a ... I want ... So that ...”
- Acceptance Criteria
- Prototype
- User Interface design
- Other text/images/content to provide context

Acceptance criteria

- Tell you the story is finished Acceptance tests tell you the system is working
- A story can have one or multiple AC's
- It informs the criteria to the tests that will be implemented or executed
- Common format:
 - Give **<context>**
 - When **<event>**
 - Then **<outcome>**

Acceptance criteria

ThoughtWorks[®]
As an Internet Banking customer
I want to see the list of my
accounts
so that I can choose to see
more details of a particular
account

Alternate path

Alternate path

Bad path

Given the customer has one transaction account and one credit account

When they have completed logging in

Then the screen should show the names and numbers of the two accounts sorted in account number order

Given the customer has just one transaction account

When they have completed logging in

Then the screen should show the name and number of the account

Given the customer has no accounts

When they have completed logging in

Then the screen should show a message stating that no accounts are available

Given the customer has more than 20 accounts

When they have completed logging in

Then the screen should show the first 20 accounts (in account number order) only

Given the customer has some accounts

And they have completed logging in

When the system cannot retrieve the account details

Then the screen should show an error message with associated code

Acceptance criteria

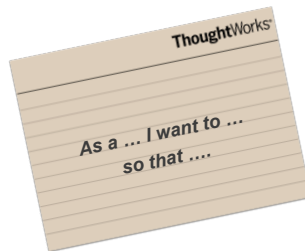
- Tell you the story is finished Acceptance tests tell you the system is working
- A story can have one or multiple AC's
- It informs the criteria to the tests that will be implemented or executed
- Common format:
 - Give **<context>**
 - When **<event>**
 - Then **<outcome>**

What is *not* in a user story

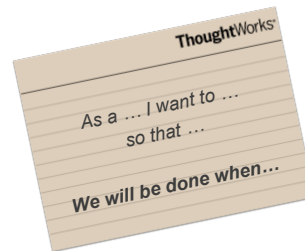
- Unit and Integration tests
- Functional/Acceptance tests
- Documentation – good documents are like vacation pictures
- Specifications

Story details at the right time

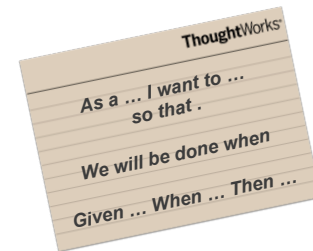
Vision Workshop



Iteration Planning

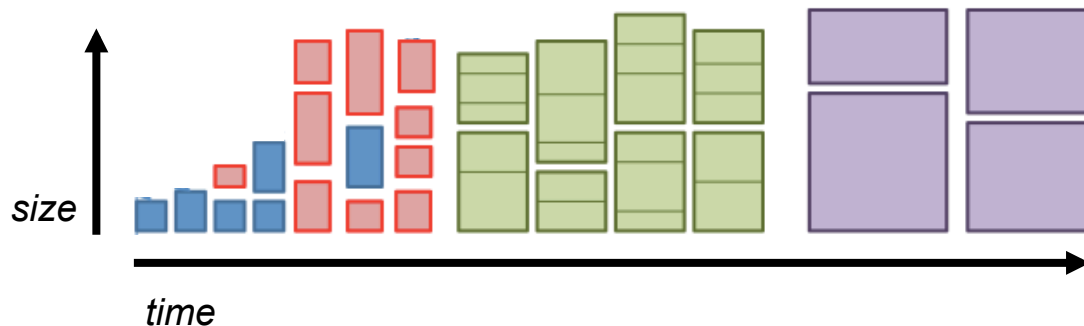


Just-in-time for development



Why just-in-time?

- Reduces potential wastage
- Provides flexibility to change, prioritize
- Enables learning from delivery
- Tighter feedback loop between business and the delivery team



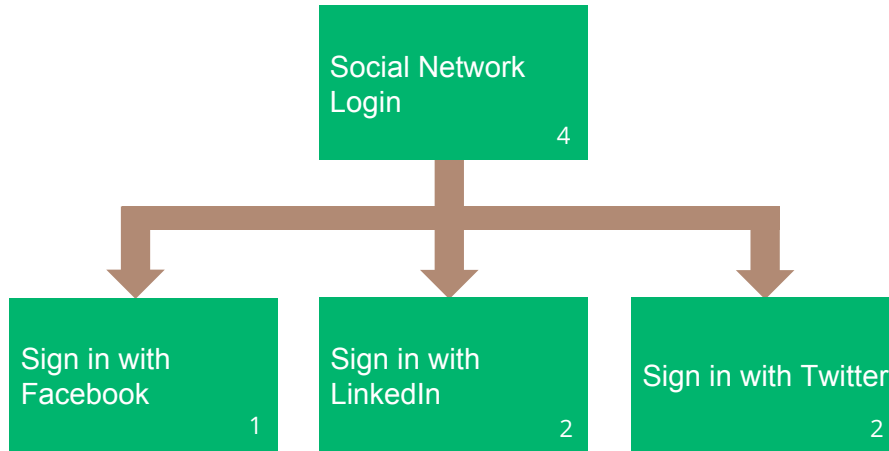
Story Breakdown



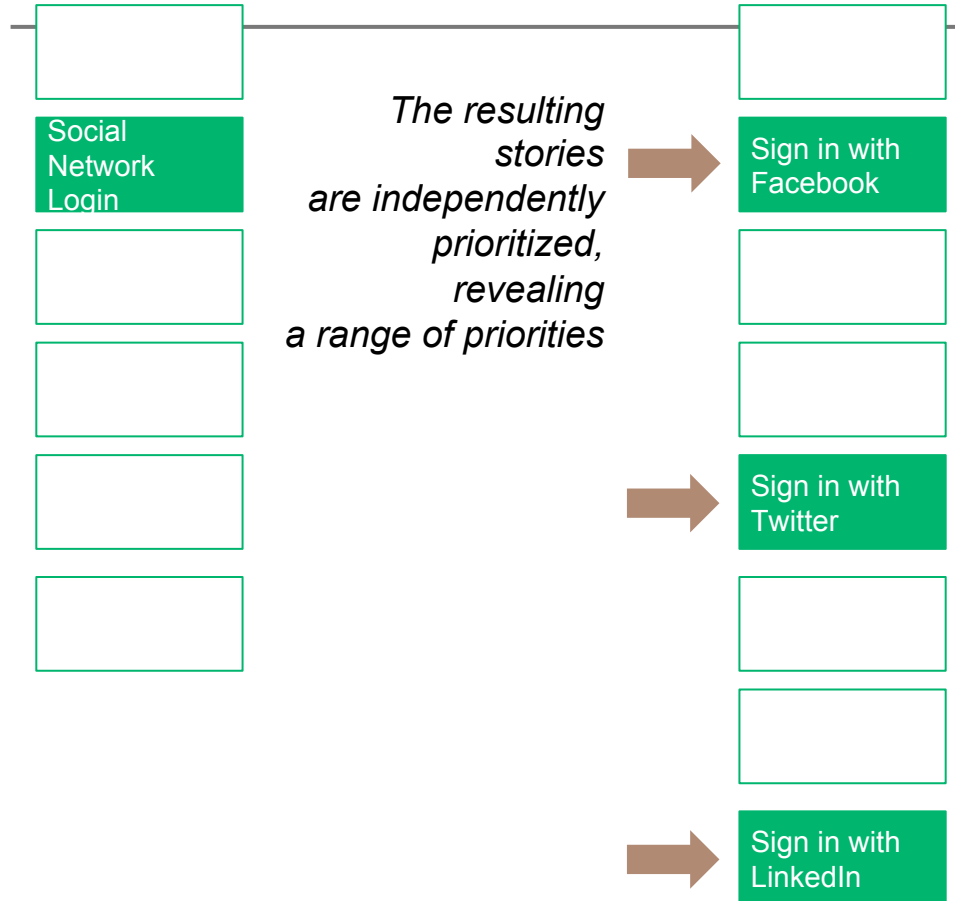
Story breakdown

- Identify opportunities to split individual stories into one or more smaller independent stories, in order to:
 - Make stories an appropriate size for delivery
e.g.. small enough that they are easily understandable, estimable and will give a good indication of progress
 - Identify differences in priority
to ensure only the highest priority work is completed first

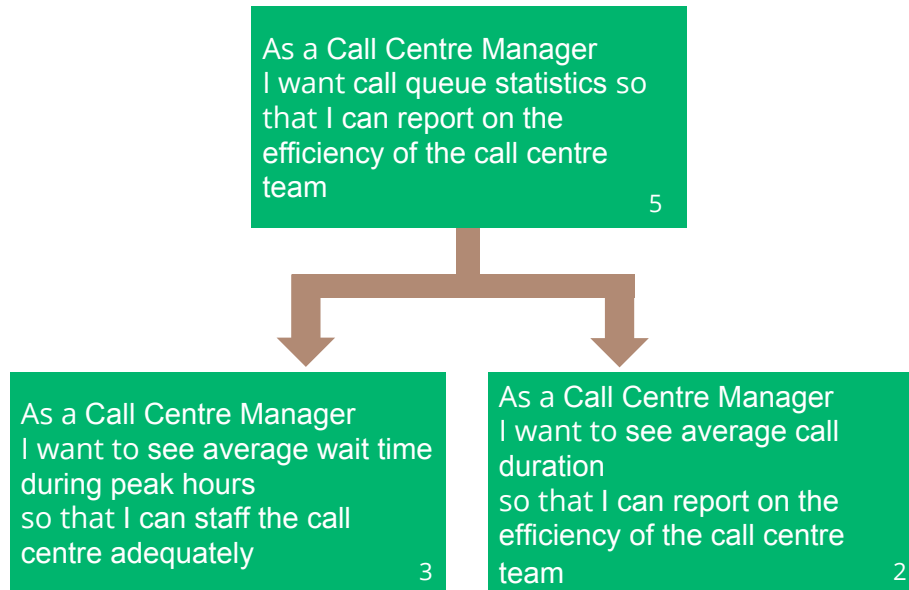
Splitting by feature



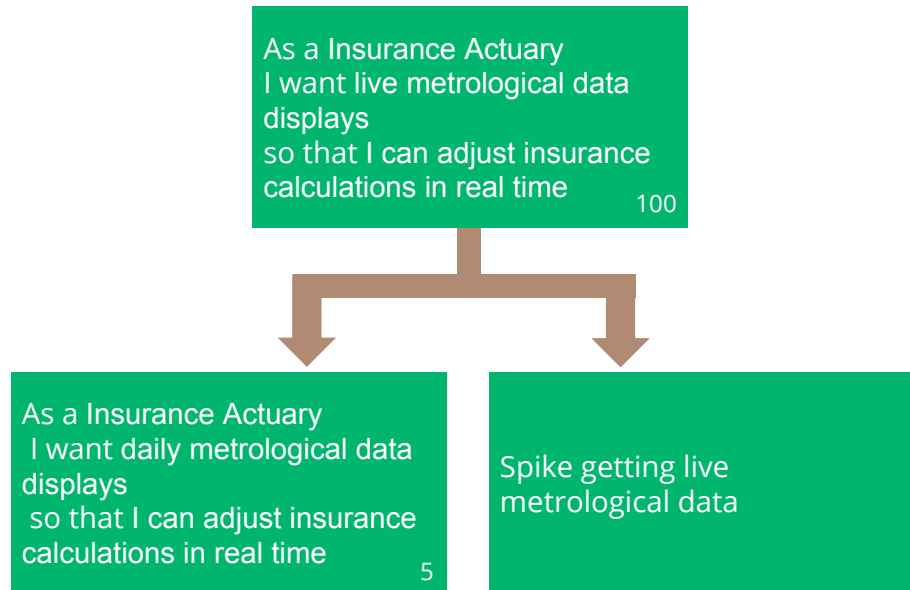
Splitting by priority



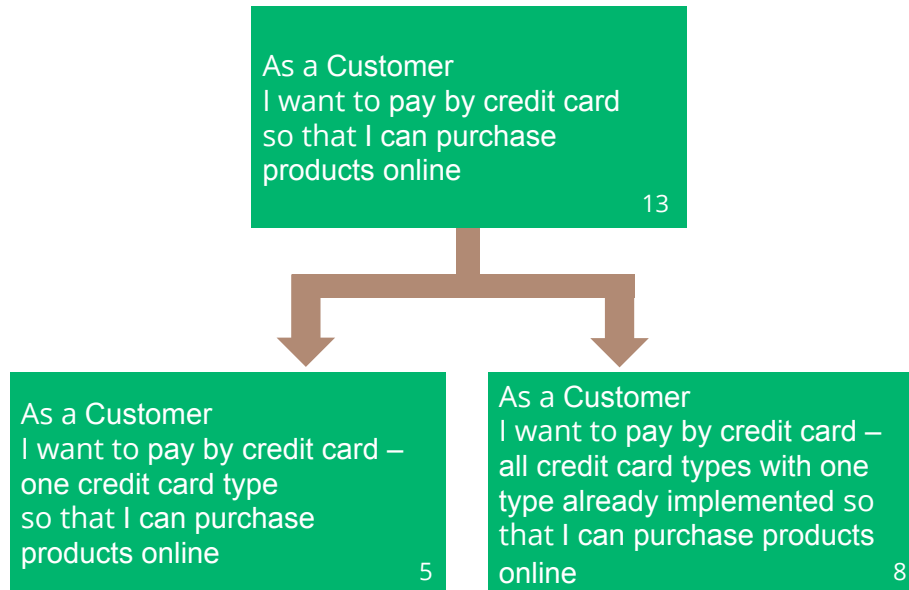
Splitting by value



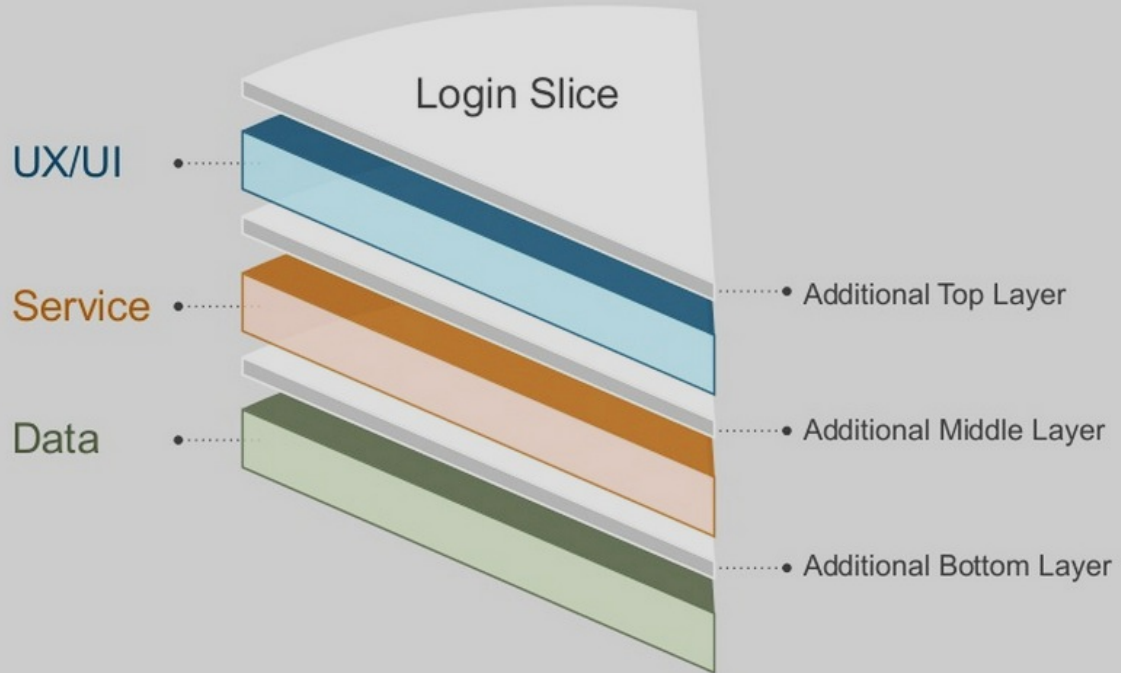
Splitting by risk



Splitting dependencies



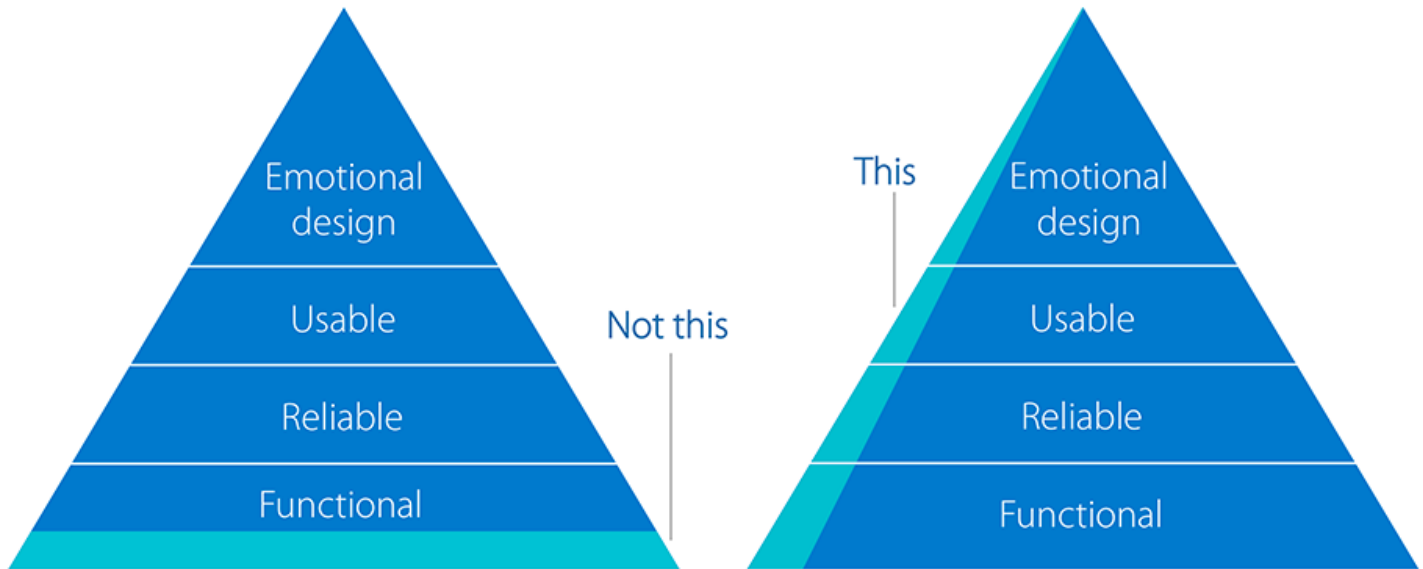
Slicing vertically



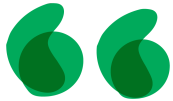
Slicing vertically



Minimum Viable Product



Story Slicing



Slice your stories like you slice your cake...

...Vertical slices across all layers.

This way, when something is done, anyone can see how the slice will fit in the bigger picture



Things to Remember When Writing User Story



Six DO's for Writing User Stories

1. Make sure there is a business goal for the story
2. Avoid passive language
3. Narrative should match the Acceptance Criteria
4. Clear Give, When, Then
5. Don't forget the sad path
6. Keep it simple!

Think about your business user and goal

- * User story should be written from the end user perspective
 - * helps to keep in mind who the functionality for
 - * Focus is on the business goal

Avoid passive/unclear language

- * Use clear direct language when writing the narrative and the acceptance criteria
- * For example:
 - * *Should, Could, Usually, Maybe*
 - * Unclear wording :
 - * There should be an error response
 - * Should return the details from before...

Narrative = AC

- * Any details of what should happen as part of a user story should also be reflected in the Acceptance Criteria
- * This is important to make sure that all details and functionality that are called out are addressed, implemented and testable

Clear ACs

- * Clear concise statements
- * Avoid multiple <WHEN>s in one AC
- * Avoid multiple <THEN>s in one AC

Don't forget Sad Path

- * A lot of times when writing a story you get focused on how the story should work = Happy Path
- * A story is not 'complete' until you address all the possible scenarios
 - * What happens if there is an error in the input?
 - * What happens if the service is unavailable?
 - * Asking questions like this can also open up discussion and/or underlying NFR needs

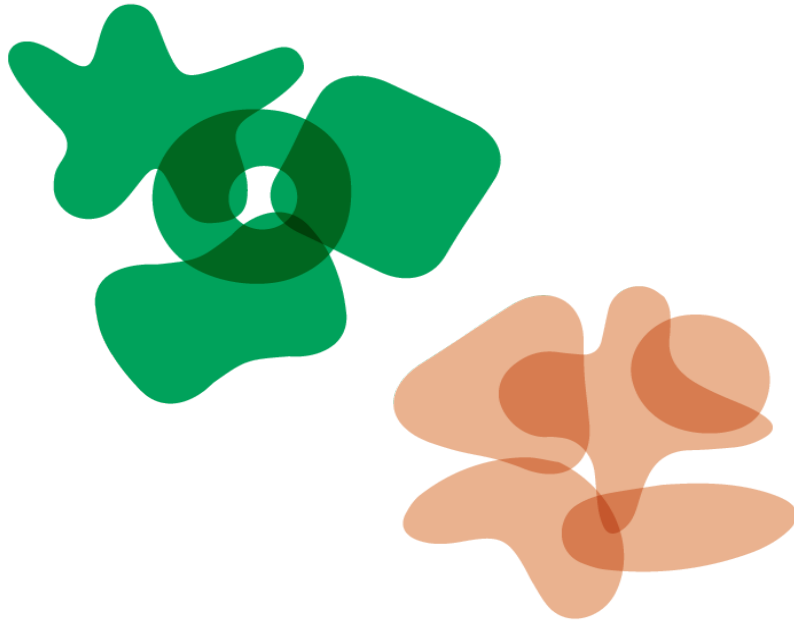
Keep it Simple

- * A lot of times when writing a story you get focused on how the story should work = Happy Path
- * A story is not 'complete' until you address all the possible scenarios
 - * What happens if there is an error in the input?
 - * What happens if the service is unavailable?
 - * Asking questions like this can also open up discussion and/or underlying NFR needs

This is not a static template

- * Keep that as the project and team changes so will the user stories
- * Information that was needed at one point may not be necessary any longer
- * What information is most important to see in a user story?
 - * Where do you focus? The whole card? Only the AC? Is the Background important to up?
- * Goal is to have anyone (in or outside the team) be able to read a story card and understand what it is about

You've Got Your Stories, How Do They Fit Together?



What is Story Mapping?

A user story map arranges user stories into a useful model to help you:

- ...understand the functionality of the system.
- ...identify holes and omissions in your backlog.
- ...effectively prioritize and groom your backlog so that value is delivered every iteration.

Map it Out

Drew

Drive to Bank

Use the ATM

See Account
Activity

As a retail customer
I want to see my balance
So that....

As a retail customer
I to see my balance in
my secondary accounts
So that....

As a retail customer
I want to see my
transaction activity
So that....

Cancel
Transaction

Drive Home

Withdrawal
Cash

As a retail customer
I want a custom amount
withdrawal option
So that....

As a retail customer
I want a standard \$20
withdrawal option
So that....

As a retail customer
I want a standard \$40
withdrawal option
So that....

As a retail customer
I want to select the
account to w/d from
So that....

Get Receipt

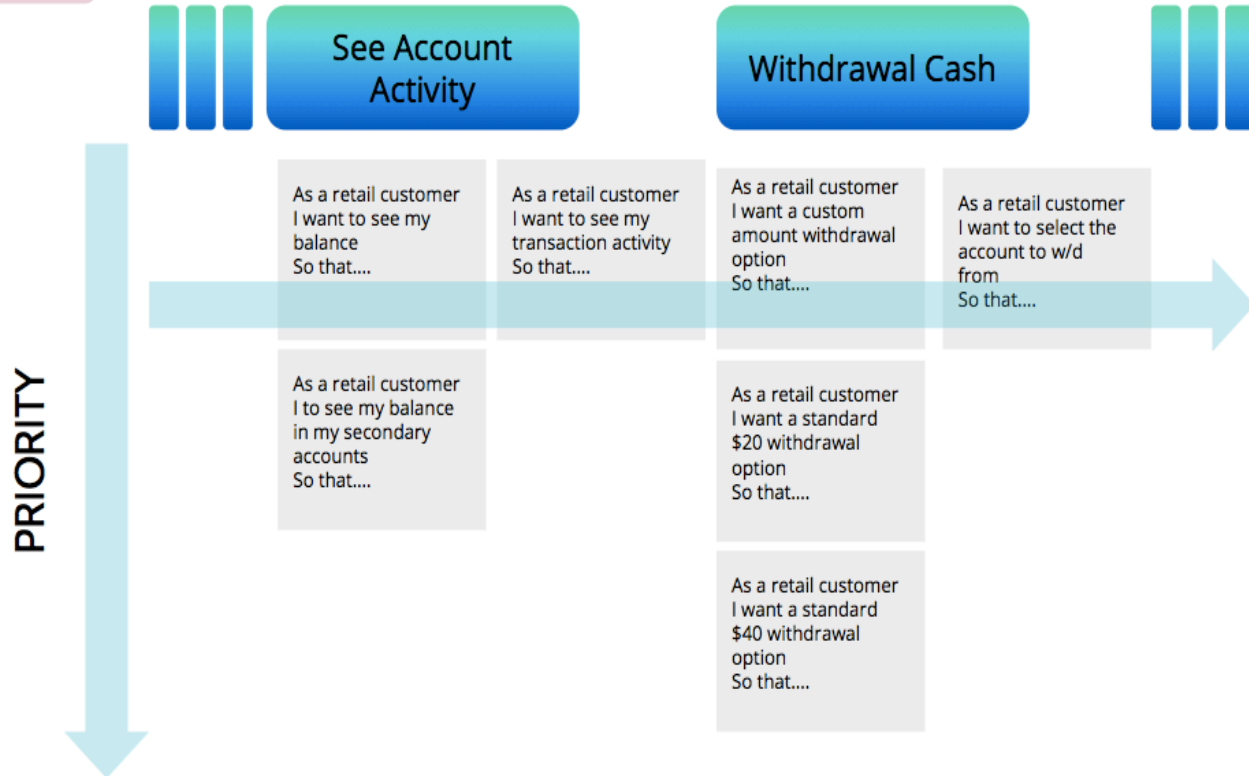
Drive to Club

STEP

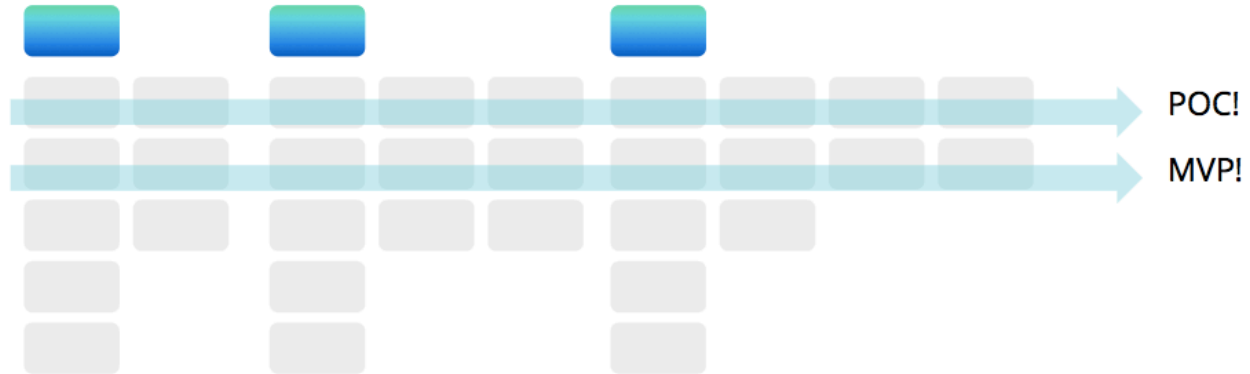


ORGANIZE BY PRIORITY

Drew



WALK THE MAP



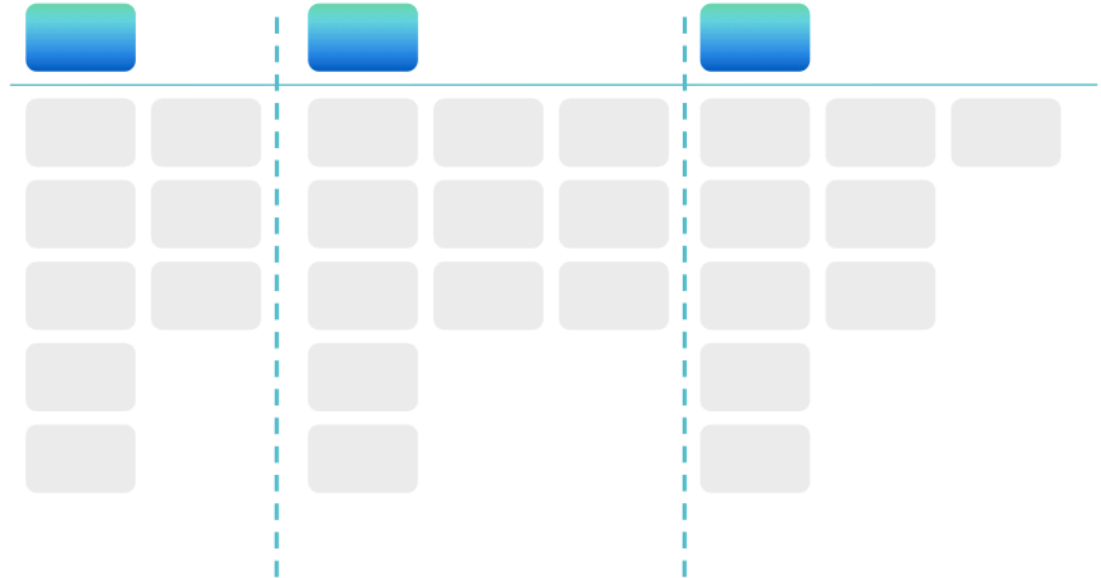
- Walking the map helps to determine your product backlog.
- Do highest value, highest priority items first.

STORY MAPS ARE DYNAMIC.

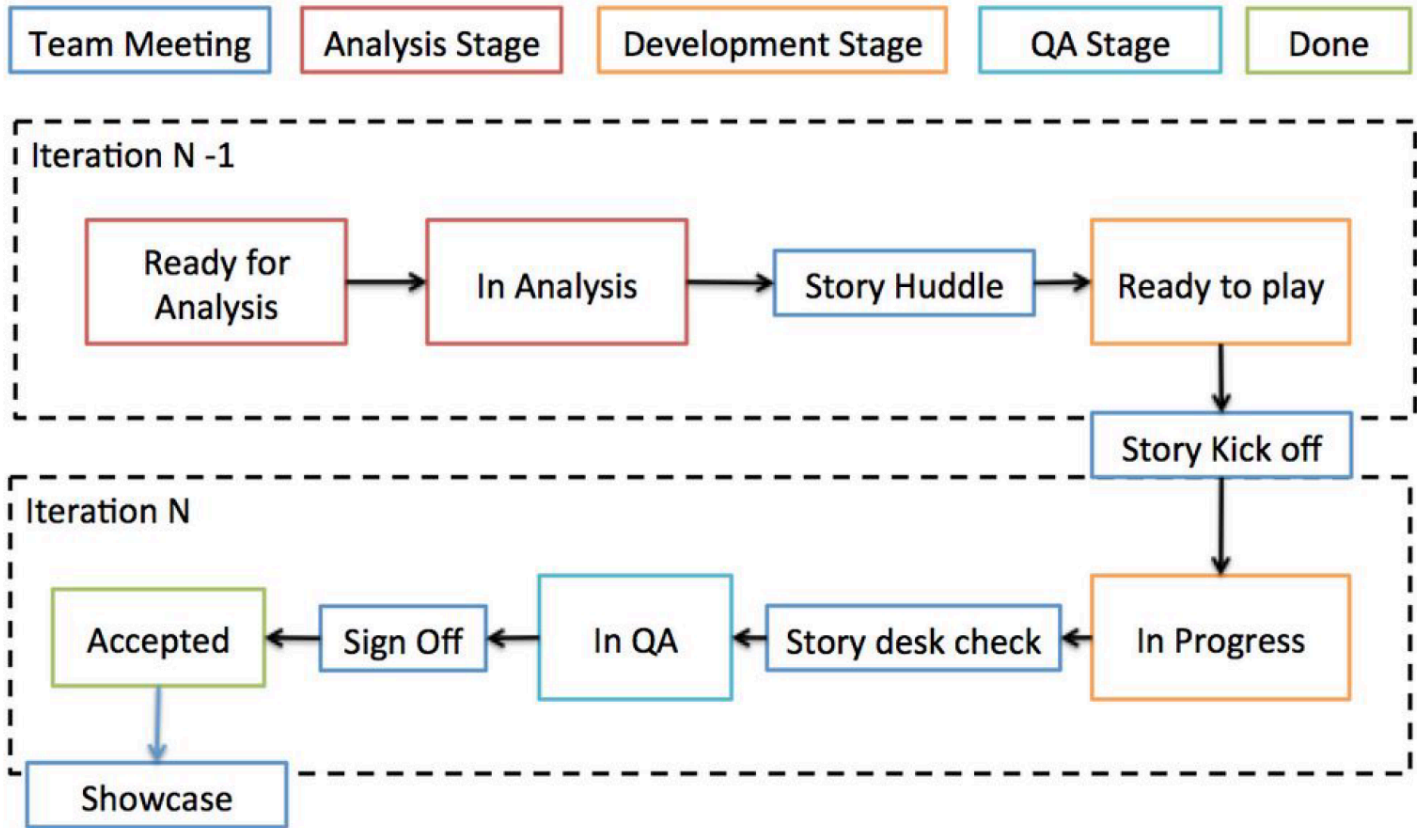
HEY! A BACKLOG!

FLAT BACKLOG

VS.



USER STORY LIFECYCLE



Questions?

ThoughtWorks®

THANK YOU

ThoughtWorks®