Unifying User stories, Use cases, Story maps. The talk.

Alistair Cockburn

Co-author, Agile Manifesto
Author: Writing Effective Use Cases

https://alistaircockburn.com/Articles/Unifying-user-stories-use-cases-story-maps-talk

What's the problem?

User stories, use cases & story maps compete for attention
of the same people
at the same time

What's the fix?

Learn to use them as partially compatible tools
in a larger toolbox.

Take advantage of each of them.

First: What are they?

User story:	A tag for what a user considers a "sign of progress" on system development
Use case:	An enumeration of all the ways for a user to achieve a goal (or fail)
Story map:	A 2D card layout showing processes L-to-R and priorities vertically down

What is a user story? [Kent Beck]

1) A short phrase or sentence that captures what a user wants. Anything they can notice (including speed) counts.
2) Not intended as a complete spec, lives in a conversation between a user and a developer. They discuss, the developer programs, shows to the user, revises, shows, until it's good.
3) * Should fit into one iteration or sprint.
4) Intended for high-collaboration environments.

Pay for goods using stored credit card

As a client, I want to pay for the goods in my basket using a stored credit card, so that I don't have to enter all the card details again.

Collect 9-digit zip code (not 4)

What is an epic? [Mike Cohn]

1) A user story, except it won't fit into an iteration or sprint.

* Calling something an 'epic' implies you will need to break it down for development.

Book all hotels, cars and flights for a vacation.

Construct and send marketing campaign for new client.

What is a use case? [Ivar Jacobson]

1) A special writing format to describe all the interactions needed for a user to achieve a goal.
2) Written with full sentences, failure conditions, how those failures are patched up (or not), and what happens at the end.
3) A full spec of the *behavior* of the system with respect to that user goal, it references back-end and external systems.
4) Does *not* detail the data, UI, performance, security needs.

* Typically needs to be broken into slices for development, that is: an *epic*.

(Sample use case)

Place an order (Sea-level goal for Clerk)

Main scenario:

1. Clerk identifies customer, item and quantity.
2. System accepts and queues the order.

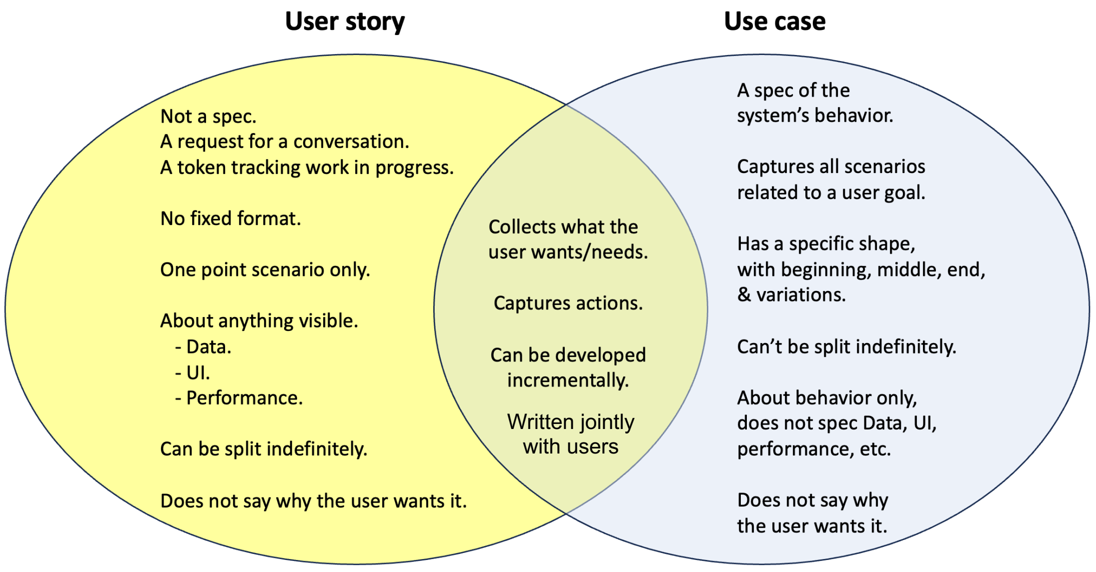
Extensions:

1a. Low credit & Customer is 'Preferred':
System gives them credit anyway.

1b. Low credit & not 'Preferred' customer:
Clerk accepts only prepayment.

2a. Low on stock: Customer accepts rain-check:
Clerk reduces order to available stock level.

User stories & use cases are *different*



User story

Not a spec.
A request for a conversation.
A token tracking work in progress.

No fixed format.

One point scenario only.

About anything visible.
- Data.
- UI.
- Performance.

Can be split indefinitely.

Does not say why the user wants it.

Collects what the user wants/needs.

Captures actions.

Can be developed incrementally.

Written jointly with users

Use case

A spec of the system's behavior.

Captures all scenarios related to a user goal.

Has a specific shape, with beginning, middle, end, & variations.

Can't be split indefinitely.

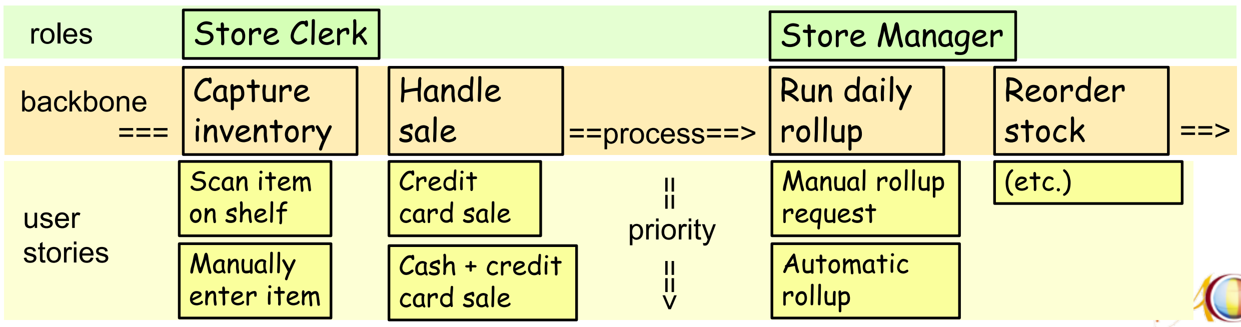
About behavior only, does not spec Data, UI, performance, etc.

Does not say why the user wants it.

Alistair Cockburn, 2024

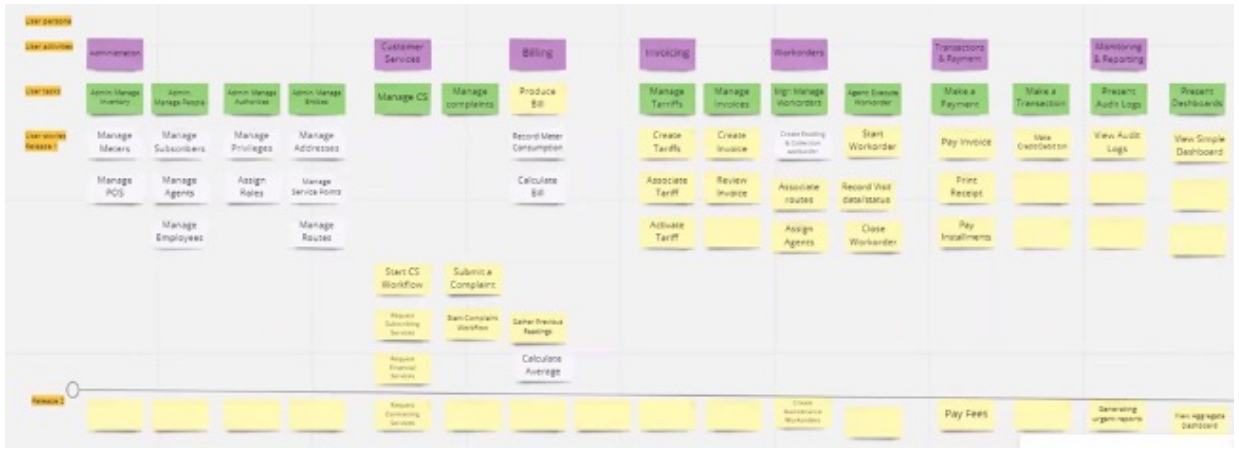
What is a story map? [Jeff Patton]

1) A 2-dimensional grid of user stories and epics.
2) Each type of user gets their own column.
   The top rows show user tasks to complete a business process;
   Each column has all user stories needed to deliver the epics.
3) Intended for high-collaboration environments.

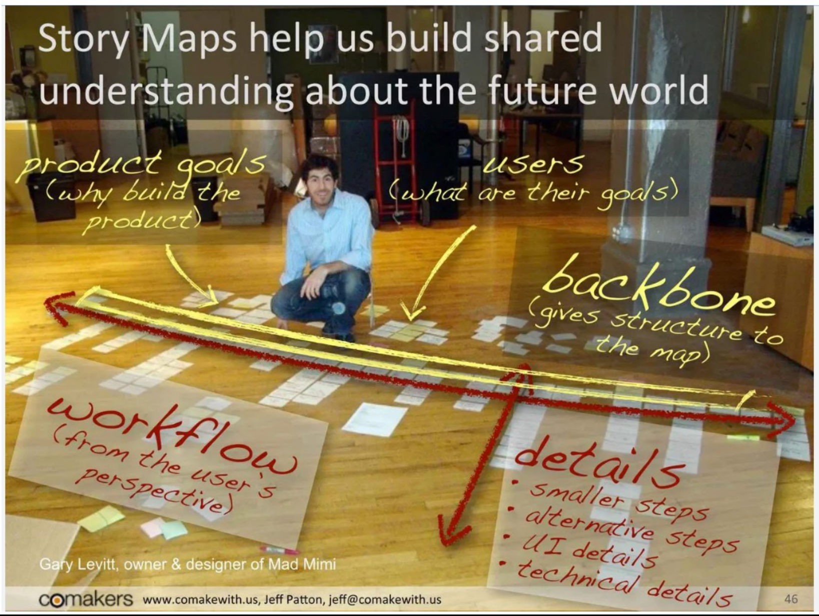


roles	Store Clerk			Store Manager		
backbone ===	Capture inventory	Handle sale	==process==>	Run daily rollup	Reorder stock	==>
user stories	Scan item on shelf	Credit card sale	=|| priority	Manual rollup request	(etc.)	
	Manually enter item	Cash + credit card sale	||=>	Automatic rollup		

(Sample story map)



(Sample story map2) [courtesy Jeff Patton]


Story Maps help us build shared understanding about the future world
product goals (why build the product)
users (what are their goals)
backbone (gives structure to the map)
workflow (from the user's perspective)
details
• smaller steps
• alternative steps
• UI details
• technical details
Gary Levitt, owner & designer of Mad Mimi
comakers www.comakewith.us, Jeff Patton, jeff@comakewith.us
46

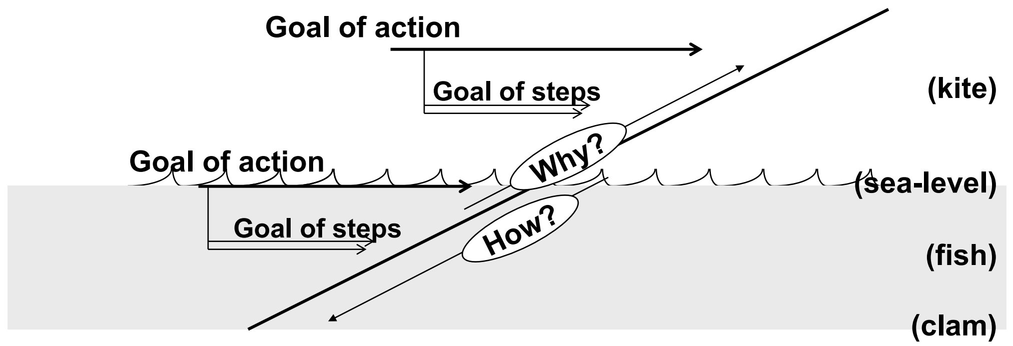
Reminder: What are they?

User story:	**A tag for what a user considers a "sign of progress" on system development**
Use case:	**An enumeration of all the ways for a user to achieve a goal (with failures)**
Story map:	**A 2D card layout showing processes L-to-R and priorities vertically down**

8 concepts needed to do well with any of them:

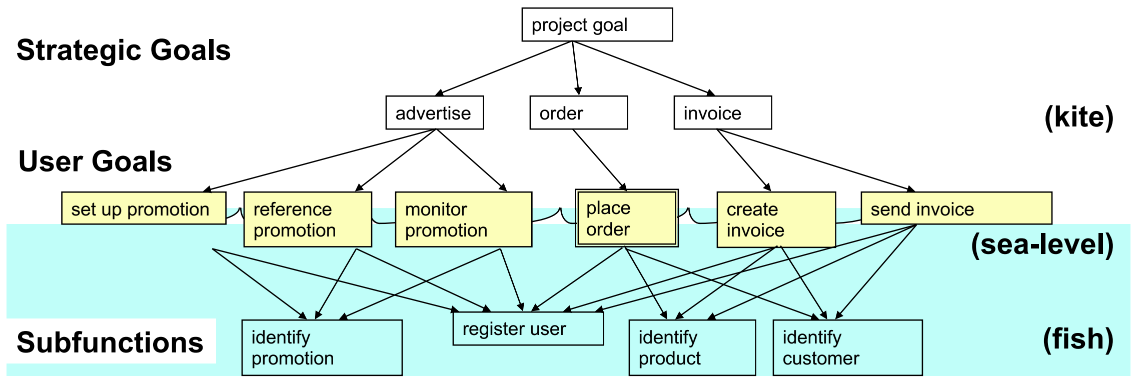
1. Verbs imply durations.
2. Decompose verbs into 'smaller' (shorter duration) verbs.
3. Manage precision.
4. Decompose everything, not just the verbs.
5. Write jointly, business & dev.
6. Write from the user's perspective.
7. Write just the needs, not the encyclopedia.
8. Sacrifice perfection for readability.

Verbs imply durations. Decompose verbs into 'smaller' verbs.



The action verb is 'higher' than the steps.
They sit on a gradient

Strategic goals, user tasks, subfunctions link together as a graph.



The 'sailboat' image.
User tasks are at sea level

8 concepts needed to do well with any of them:

1. Verbs imply durations.
2. Decompose verbs into ‘smaller’ (shorter duration) verbs.
3. Manage precision.
4. Decompose everything, not just the verbs.

---

5. Write jointly, business & dev.
6. Write from the user’s perspective.
7. Write just the needs, not the encyclopedia.
8. Sacrifice perfection for readability.

Decompose verbs:

For use cases:

Don’t decompose below fish level.
Keep the use case shape (main + extensions)

For user stories:

Decompose down to clam level as needed.
Can decompose almost indefinitely.

Decompose use cases into user stories:

1. Choose the thinnest full transaction as slice 1.
2. Choose any action/extension that fits an iteration.
3. Subset any action/extension until it's small enough.

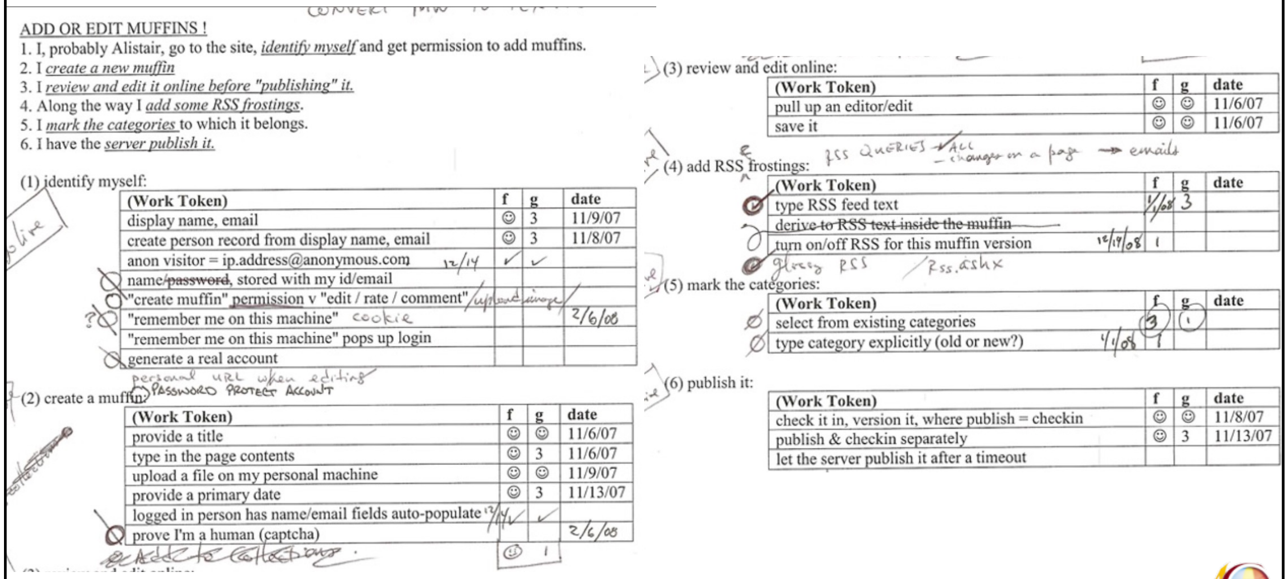
UC 7: "Register for Courses" [*Patterns for Effective Use Cases*, Adolph-Bramble)]

System: Course Enrollment System Goal level: User (sea) –level goal

[1] 1. Student requests to construct a schedule. **("base function")**
[1] 2. The system prepares a blank schedule form. **("added function")**
[2] 3. The system gets available courses from the Course Catalog System.
[3] 4. Student selects up to 4 primary and 2 alternate course offerings.
[4,5] 5. For each course, the system verifies that the Student has the necessary prerequisites, adds the Student to the course, marking Student as "enrolled" for that course in the schedule.
[1] 6. The Student indicates the schedule is complete, the system saves it.

Extensions:
[6] 1a. *Student already has a schedule*: System brings up the current version of the Student's schedule for editing instead of creating a new one.
[7] 1b. *Current semester is closed and next semester is not yet open*: System lets Student look at existing schedules, but not create new ones.
[8] 3a. *Course Catalog System does not respond*: The system notifies the Student and the use case ends.
[5] 5a. *Course full or Student has not fulfilled all prerequisites*: System disables selection of that course and notifies the Student.

Decompose use cases also by action variations:



ADD OR EDIT MUFFINS !

1. I, probably Alistair, go to the site, *identify myself* and get permission to add muffins.
2. I *create a new muffin*
3. I *review and edit it online before "publishing" it.*
4. Along the way I *add some RSS frostings*.
5. I *mark the categories* to which it belongs.
6. I have the *server publish it.*

(1) identify myself:

(Work Token)	f	g	date
display name, email	☺	3	11/9/07
create person record from display name, email	☺	3	11/8/07
anon visitor = ip.address@anonymous.com 12/14	✓	✓	
name/~~password~~, stored with my id/email			
"create muffin" permission v "edit / rate / comment" / upload image /			
"remember me on this machine" cookie			2/6/08
"remember me on this machine" pops up login			
generate a real account			

personal URL when editing

PASSWORD PROTECT ACCOUNT

(2) create a muffin:

(Work Token)	f	g	date
provide a title	☺	☺	11/6/07
type in the page contents	☺	3	11/6/07
upload a file on my personal machine	☺	☺	11/9/07
provide a primary date	☺	3	11/13/07
logged in person has name/email fields auto-populate 12/14	✓	✓	
prove I'm a human (captcha)			2/6/08
Add to Collections	☺	1	

(3) review and edit online:

(Work Token)	f	g	date
pull up an editor/edit	☺	☺	11/6/07
save it	☺	☺	11/6/07

(4) add RSS frostings: RSS QUERIES ALL changes on a page → emails

(Work Token)	f	g	date
type RSS feed text	1/1/08	3	
~~derive to RSS text inside the muffin~~			
turn on/off RSS for this muffin version 12/19/08	1		
glossy RSS /Rss.ashx			

(5) mark the categories:

(Work Token)	f	g	date
select from existing categories	3	1	
type category explicitly (old or new?) 1/1/08	1		

(6) publish it:

(Work Token)	f	g	date
check it in, version it, where publish = checkin	☺	☺	11/8/07
publish & checkin separately	☺	3	11/13/07
let the server publish it after a timeout			

Decompose data, UI, performance, security:
(not in the use cases)

Personal information:
- Name:
  - First name, Middle initial, Last name
- Address:
  - Street, City, Zip code, State
- Phone number:
  - Home, Business, Cell

Payment details:
- Credit card:
  - Name, Number, Expiration

8 concepts needed to do well with any of them:

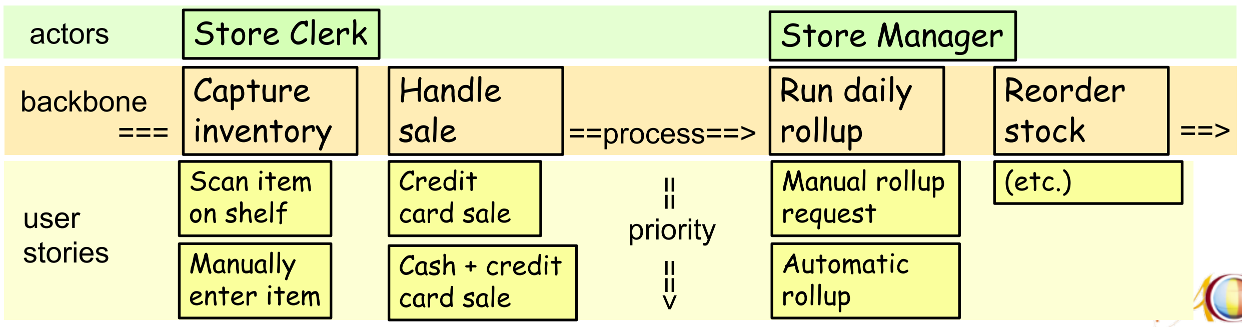
1. Verbs imply durations.
2. Decompose verbs into ‘smaller’ (shorter duration) verbs.
3. Manage precision.
4. Decompose everything, not just the verbs.

---

5. Write jointly, business & dev.
6. Write from the user’s perspective.
7. Write just the needs, not the encyclopedia.
8. Sacrifice perfection for readability.

A story map is a mix of use case & user stories

Each type of user gets their own column (the actors).
The top rows show the overall process
(use case main success scenario).
Each column has all user stories needed to deliver the epics
(slicing epics, use cases, failures, data, user stories)



Reprise: What are they good for?

User story: A *tag*. Useful for tracking where the request is during development up to delivery.

Use case: *Tells a story* easily read across the org.
A context around specific requests.
A structure for discovering oddball cases.

Story map: A *conversation-holder* showing both large-scale context & fine-grained stories.

https://alistaircockburn.com/Articles
/Unifying-user-stories-use-cases-story-maps-talk

Unifying
User stories,
Use cases,
Story maps. The talk.

Alistair Cockburn
Co-author, Agile Manifesto
Author: Writing Effective Use Cases