"That's NOT what I meant when I said we need to decompose our computer system!"

Comp 3120/8110

Software Development Management Week 6 Lecture 1 – Functional Decomposition and Writing User Stories in detail

Assignment #1

All 50 groups submitted on time – well done!

Peer Assessment has been extended to tomorrow. The new deadline is Wednesday, 31 March at 12noon AEST.

Note that after this time peer assessment will not be available.

Assignment #2

Assignment 2 - Research Report - 15% of total marks
Small groups of 4-5 students are required to research an assigned topic and produce a fully referenced research paper related to that topic. Ideally, reports will be between 4-5, A4 pages. Diagrams are permitted, but dot points should be used sparingly.

Reports are expected to include at least three appropriate references which may be drawn from a mixture of reputable journals (e.g. CIO magazine, Forbes, Harvard Business Review), text books, corporate websites, education sites and professional blog sites. Citation and referencing must follow an appropriate standard and be used correctly.

Papers are to be delivered in PDF format via the appropriate wattle submission activity.

Which topic should you research?
Your assignment group will be the same as for Assignment 1 subject to any minor changes necessitated by student movements, which may necessitate your group being rearranged. Your group is to research exactly one of these topics listed below; you will be assigned the topic during Week 5.

Topics
Project and professional ethics
Leadership styles and team motivation
Communicating with stakeholders and managing their expectations
Communication and the delivery of bad news

Assessment value	15% of the overall course mark
Date Due	Submission is due 5pm Friday, Week 7 Peer assessment is due 9am Tuesday, Week 8
Topic	Assigned in Week 6.
Required information	§ A title, which clearly identifies the topic § The date of preparation of the research report § The student number of each member of the group § Executive Summary § Introduction to and background of the topic § Discussion § Conclusions § References Note – Reports may include additional headings § Provides answers to all questions posed as part of the topic
References	A minimum of 3 appropriate references.
Page count	Including references, but excluding appendices, maximum allowable A4 pages is 8. Recommended 4-5 pages, assuming 300-350 words per page. Font, line spacing and margins need to be appropriate for a management document.
File format	PDF only
Submission – Report	**GROUP submission** – i.e. a single copy per group – via the appropriate Wattle / Turnitin assignment submission activity to be provided shortly.
Submission – Peer Assessment	**INDIVIDUAL submission** – i.e. each student must submit a peer assessment via Wattle - link will be provided nearer the due date.
Expected hours	About 10 hours for **each** student in the group

Link to Assignment #2

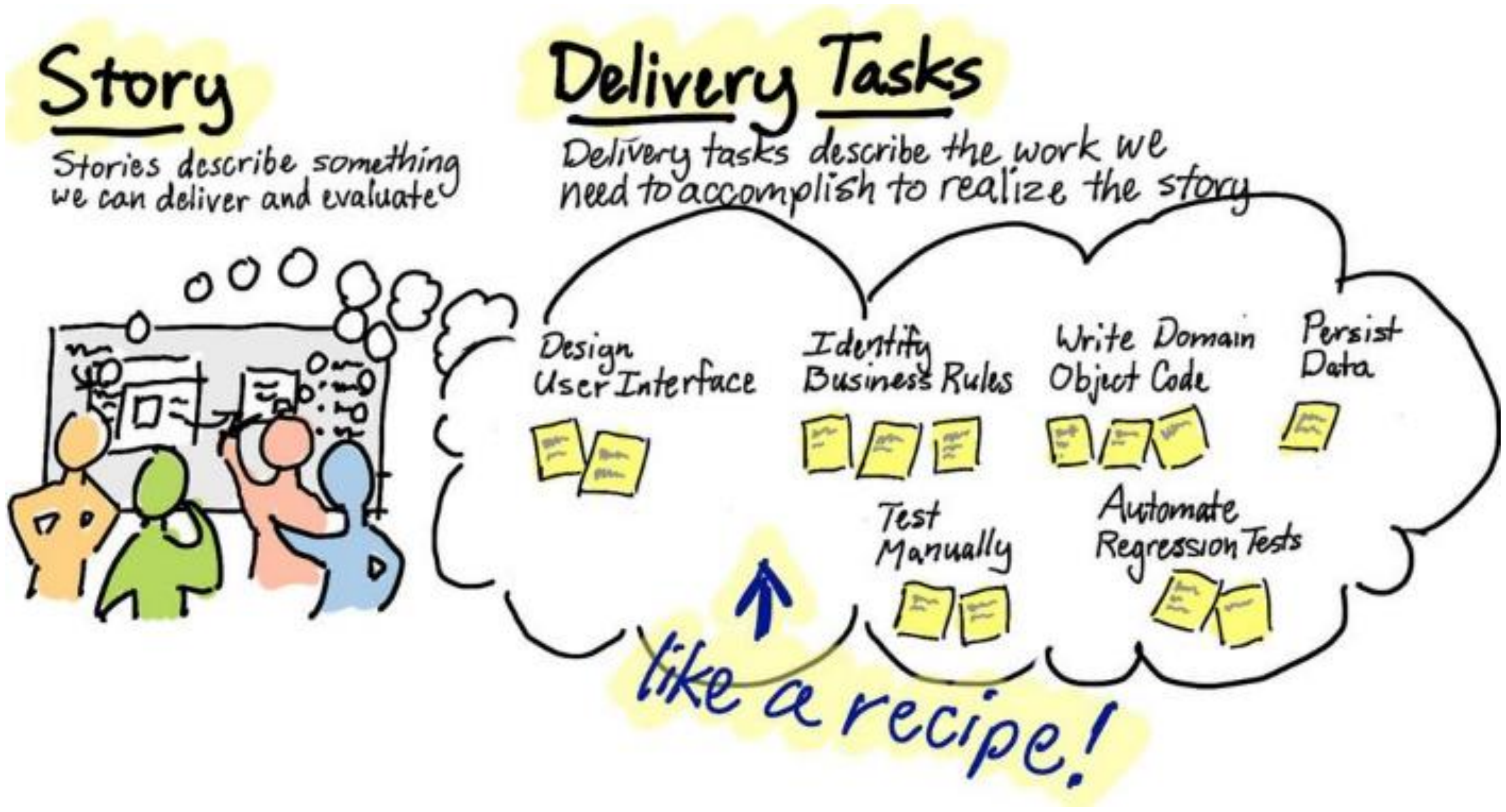
Functional Decomposition

When faced with a large complex idea or process, you can make it manageable by breaking it down, or decomposing it, into smaller, simpler pieces.

Functional Decomposition

Shared understanding between all stakeholders

Stories get broken down into **tasks**



Story
Stories describe something we can deliver and evaluate
Delivery Tasks
Delivery tasks describe the work we need to accomplish to realize the story
Design User Interface
Identify Business Rules
Write Domain Object Code
Persist Data
Test Manually
Automate Regression Tests
like a recipe!

Who **What** **Why** **Context**

Functional Decomposition

Epic versus User Story – What is the right size?

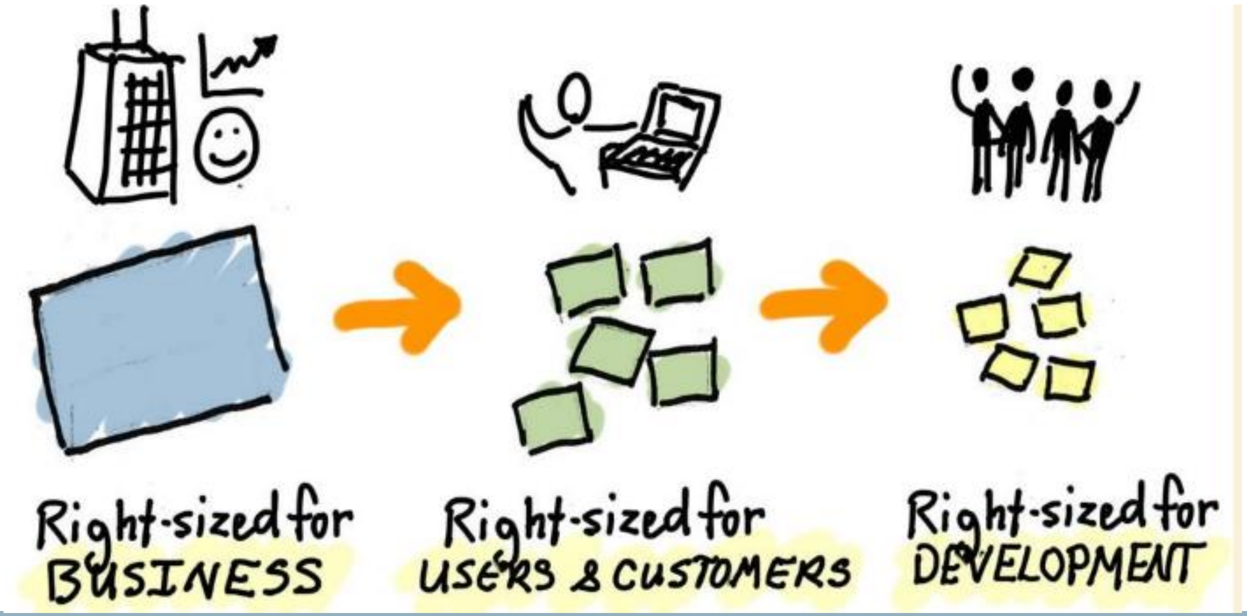
Right size depends on your perspective!

Different stakeholders have different perspectives on size.

User's perspective -- the right size is one that fulfils a need

Development team's perspective -- the right size is one that takes just a few days to build & test

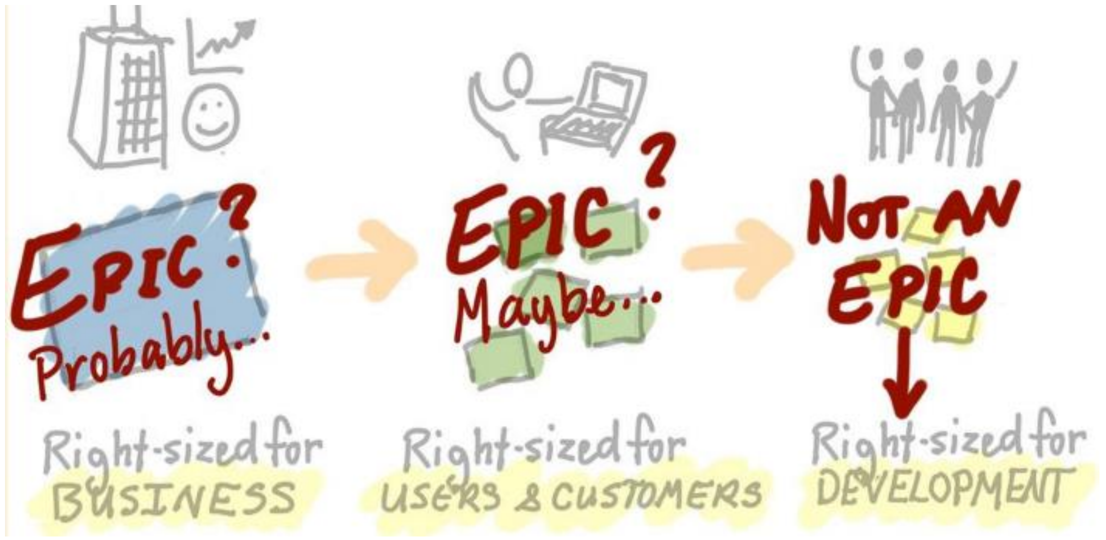
Business perspective -- the right size is one that helps a business achieve a business outcome


Right-sized for BUSINESS
Right-sized for USERS & CUSTOMERS
Right-sized for DEVELOPMENT

Australian National University

Functional Decomposition

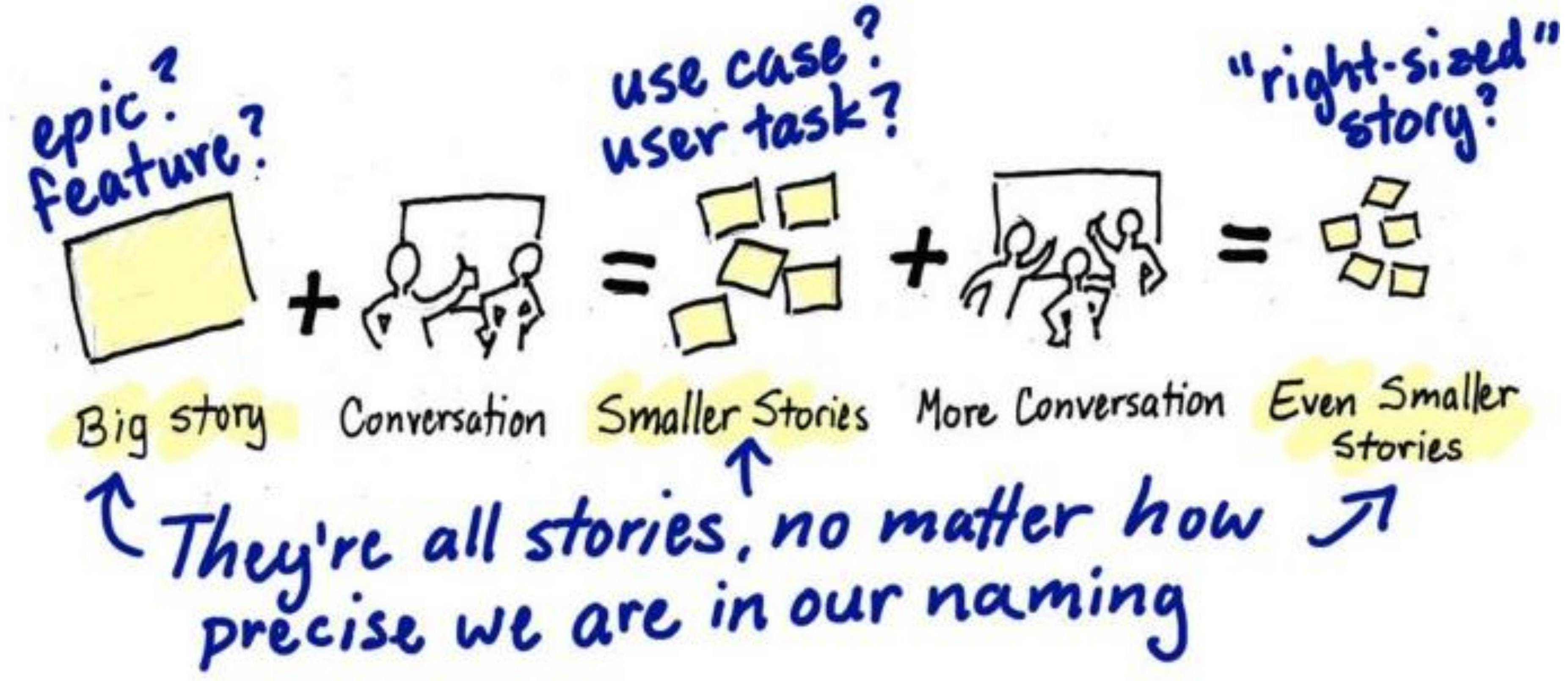
So when is it an epic and when is it a story?



Epics rather than smaller user stories should be used when

- undertaking long-term planning;
- during initial estimation as part of a business case or project initiation, when a perfect, very precise estimate is not required. Instead, a high-level estimate, provided through a high-level product backlog, is sufficient, where big user stories – epics – describe large swaths of functionality; and
- before you are ready to develop them -- **remember just-in-time decomposition is the way to go.**

Functional Decomposition



Functional Decomposition

How to break down epics into stories

The ability to split stories is an important skill for customers and developers.

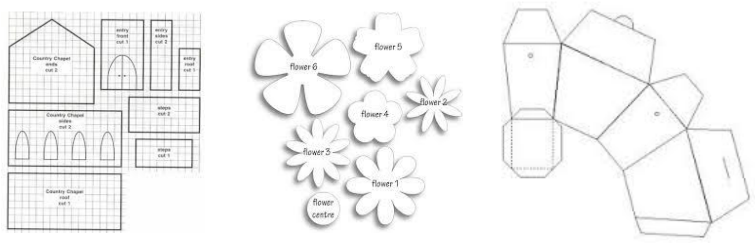
Often splitting stories **separate high-value parts from low-value parts**, allowing the team time to focus on the valuable parts of a feature. This helps to provide value by enabling development of a minimal, end-to-end solution and then filling in the rest of the solution. Frequently splitting stories is called decomposition.

Decomposition takes practice and experience. Some stories are more difficult than others to split. Discovery conversation is one of the best tools for breaking down big stories.

Remember to only decompose epics only when more detailed estimation is required

Flowchart for Splitting Epics

- Prepare the input story
- Apply the splitting pattern
- Evaluate the split



Australian National University

Functional Decomposition

What's a 'splitting pattern'?

There are many ways to split epics into the right size user story.

For example: ***Nine Patterns for Splitting Epics***

1. Workflow steps
2. Business rule variations
3. Major effort
4. Simple / Complex
5. Variations in data
6. Data entry methods
7. Defer performance
8. Operations
9. Break out a "spike"

I	Independent	Be able to develop in any sequence
N	Negotiable	Keep flexible so team can adjust what is implemented
V	Valuable	Users/customers get some value
E	Estimatable	Team must be able to use them for planning
S	Small	By the iteration, be able to be designed, coded and tested in iteration
T	Testable	Acceptance criteria and/or definition of done – leads to your test cases

However you do it, you want to end up with this for development!

There are different reasons to use different patterns.

Functional Decomposition

What's a 'splitting pattern'?

Nine Patterns for Splitting Epics

1. Workflow steps

   Build a simple end to end case first, then add the middle steps and special cases.
   For example, As a content manager, I can publish a news story to the corporate website.
   ...I can publish a news story directly to the corporate website.
   ...I can publish a news story with editor review.
   ...I can publish a news story with legal review.

2. Business rule variations

   As a user, I can search for flights with flexible dates.
   ...as "n days between x and y."
   ...as "a weekend in December."
   ...as "± n days of x and y.

3. Major effort

   Where a story has several parts, but most effort goes to the first one.
   For example, splitting a story by credit card type: most effort goes to creating the first one
   As a user, I can pay for my flight with VISA, MasterCard, Diners Club, or American Express.
   ...I can pay with one credit card type (of VISA, MC, DC, AMEX).
   ...I can pay with all four credit card types (VISA, MC, DC, AMEX).

Functional Decomposition

What’s a ‘splitting pattern’?

Nine Patterns for Splitting Epics

4. Simple / Complex

Capture the simplest version along with some acceptance criteria.
Then add variations and complexities into their own stories. For example,
As a user, I can search for flights between two destinations.
...specifying a max number of stops.
...including nearby airports.
...using flexible dates.

5. Variations in data

Build the simplest first. For example,
As a content manager, I can create news stories.
...in English. ...in Japanese. ...in Arabic. ...etc.

6. Data entry methods

Complexity is often in the UI, not the story. For example,
As a user, I can search for flights between two destinations.
...using simple date input.
...with a fancy calendar UI.

Functional Decomposition

What's a 'splitting pattern'?

Nine Patterns for Splitting Epics

7. Defer performance

Start with the easiest and provide value. Make it work, make it fast.
As a user, I can search for flights between two destinations.
...(slow - just get it done, show a "searching" animation).
...(in under 5 seconds).

8. Operations

Often stories begin with the word "manage". For example
As a user, I can manage my account.
...I can sign up for an account.
...I can edit my account settings.
...I can cancel my account.

9. Break out a "spike"

When the implementation is poorly understood a time-boxed spike to resolve uncertainty around implementation is a good approach.
For example, the story "*As a user, I can pay by credit card*" might require the team to
- Investigate credit card processing.
- Implement credit card processing (as one or more stories).

Australian National University

Functional Decomposition

From the perspective development team perspective, what size is best?
Should it be:

A day?
A week?
Three weeks?
1 point?
5 points?
10 points?
5 words?
20 words?

Functional Decomposition

From the perspective development team perspective, what size is best?
Should it be:

A day?
A week?
Three weeks?
1 point?
5 points?
10 points?
5 words?
20 words?

There is no right size!
It depends on
the team
their skills
the level of their domain knowledge
and
their velocity

BUT, generally smaller is better than larger.

The story needs to be small enough for the team to understand and to develop in a short time period which is usually less than one iteration. BUT it needs to be big enough to represent business value in its own right.

Functional Decomposition

How many stories is enough?

Teams need to create just enough high-value stories to contribute to an understanding of the project's requirements and end goal. Too many stories leads to

Bloated backlogs
Extra work (possibly mis-work) on stories
Team taking its eyes off business value
Loss of focus on the user

Functional Decomposition

How many stories is enough?

Teams need to understand how stories contribute to building and realising business goals and objectives and delivering value to stakeholders.

Stories need to be prioritised by more than just the user’s perspective. Don't overlook business, compliance, and technology.

Stories should only be written within view of the planning horizon -- the next development cycle. Let the release plan or road map drive the time frame for refining stories.

Backlog grooming lets teams refine a smaller number of stories shortly before they are needed. This approach leads to a team being more efficient and their estimates more accurate.