Armaxon

IoT made simple

Idea to Live. Product Development Approach.

Prepared for SweetTree Home Care Services

Our main challenges:
Transparency
The IT approach to building a product

There is no definition of backlog artefacts, no hierarchy
PO does not understand the sizes of backlog elements - US – tasks
No one knows team's velocity
There is no consistency in building the product – all US look a bit standalone
There is no Definition of Ready and Definition of Done
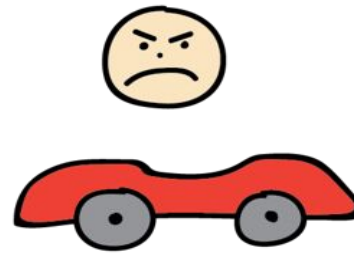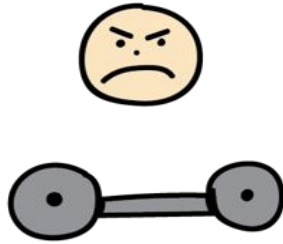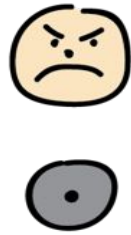**There is no product roadmap, no PSIs identified**
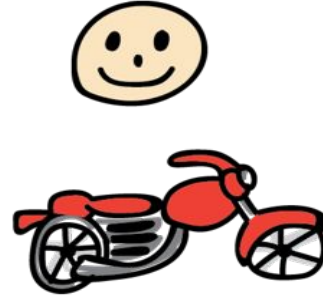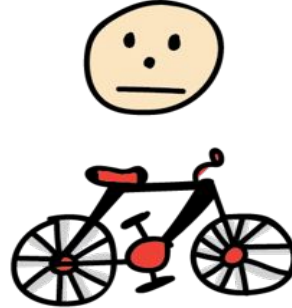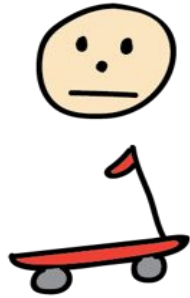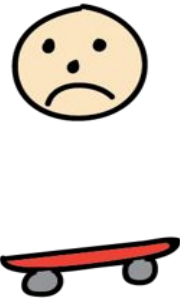**"It's an internal approach" type of comments confuse the client**

Armaxon

Product Development Approach

Not Like This!

Like This!

Armaxon

A Program Increment (PI) is a timebox during which an Agile team delivers incremental value in the form of working and tested software.

SPRINT 1   SPRINT 2   SPRINT 3   SPRINT 4   SPRINT 5

10 weeks
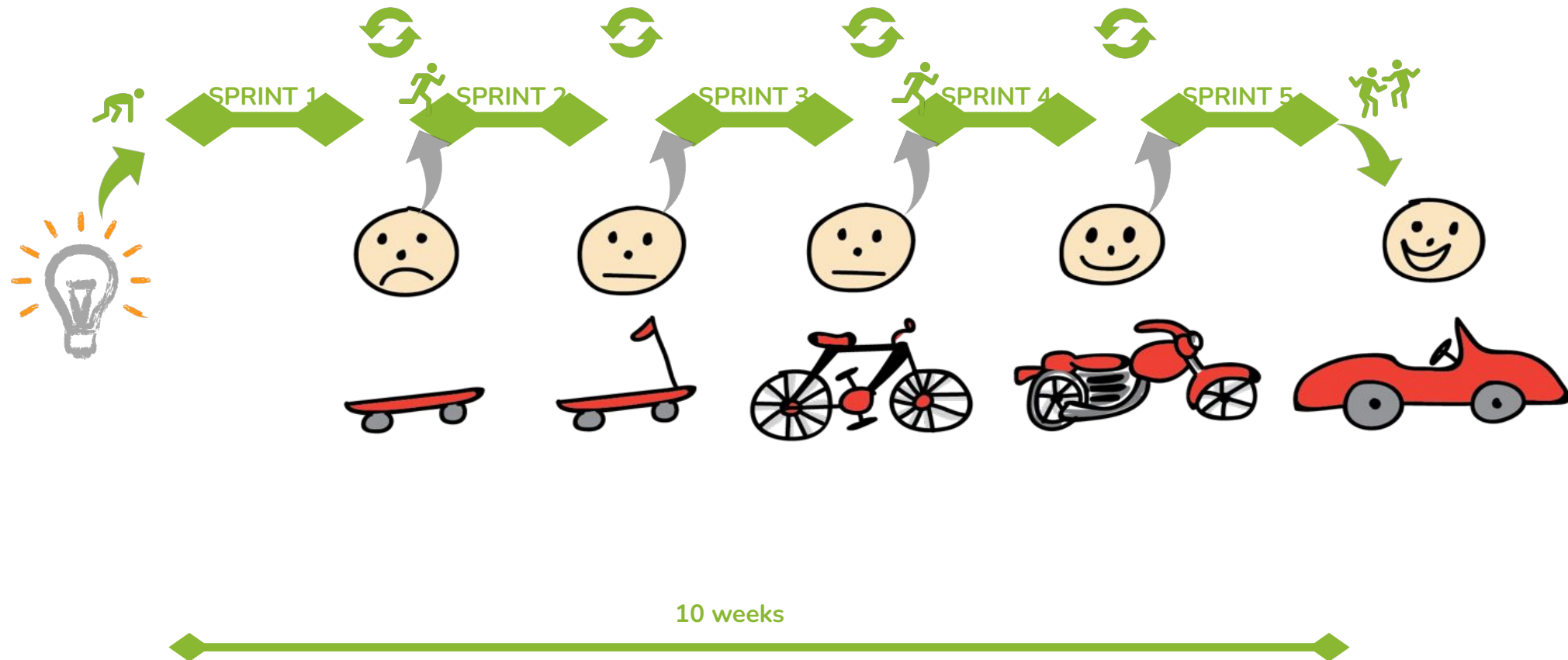
The Scrum model expects the team to bring the product or system to a potentially shippable state at the end of each Scrum sprint.

Activity	Typical Sprint									
	Wed	Thu	Fri	Mon	Tue	Wed	Thu	Fri	Mon	Tue
Daily Stand Up The team can complete the final burndown chart and individually assess what was completed and what wasn't.		✓	✓	✓	✓	✓	✓	✓	✓	✓
Individual wrap-up The team can complete the final burndown chart and individually assess what was completed and what wasn't.	✓									
Sprint Review The whole team and the product owner together to review everything that's been accomplished.	✓									
Sprint Retrospective Discuss any learnings and how the team could have performed better.	✓									
Sprint Planning While the retrospective and review is fresh in everybody's head.	✓									
Backlog Refinement Business Analyst and Product Owner prepares US for discussion. Team evaluate them, ask questions.						✓				
Development										

Armaxon

Review and planning day

Activity	Wednesday (Day 1 of a sprint)				
	8am - 9am	9am – 10am	10am - 11am	11am – 12pm	12pm – 1pm
Individual wrap-up The team can complete the final burndown chart and individually assess what was completed and what wasn't.	Team				
Sprint Review The whole team and the product owner together to review everything that's been accomplished.		Team Product Owner			
Sprint Retrospective Discuss any learnings and how the team could have performed better.			Team Product Owner		
Break					
Sprint Planning While the retrospective and review is fresh in everybody's head.					Team Product Owner

Armaxon

Backlog Management

Backlog Structure

Program backlog

Team backlog

Initiative	

Feature 1 - 3500pt
Feature 2 - 3000pt
Feature 3 - 4000pt

Epic 1 - 100pt
Epic 2 - 100pt
Epic 3 - 80pt
Epic 4 - 80pt
Epic 5 - 120pt
Epic 6 - 180pt

User story 1 - 8pt
User story 2 - 13pt
User story 3 - 8pt
User story 4 - 5pt
User story 5 - 13pt
User story 6 - 1pt
User story 7 - 3pt
User story 8 - 8pt

Armaxon
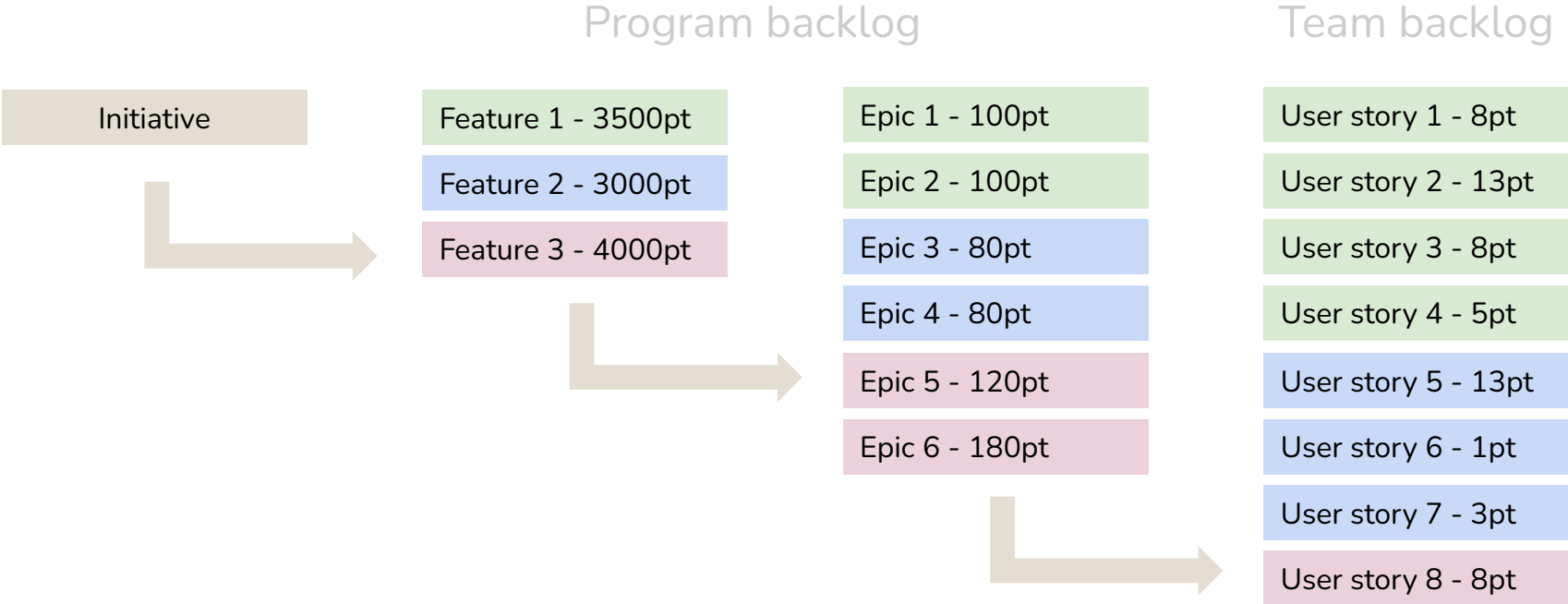
Team backlog - holds and prioritises User Stories and Enabler Stories

Who prioritises: Product Owner, Primary Stakeholders

Who defines and estimates: Product Owner, Primary Stakeholders, Business Analysts, **Developers!**
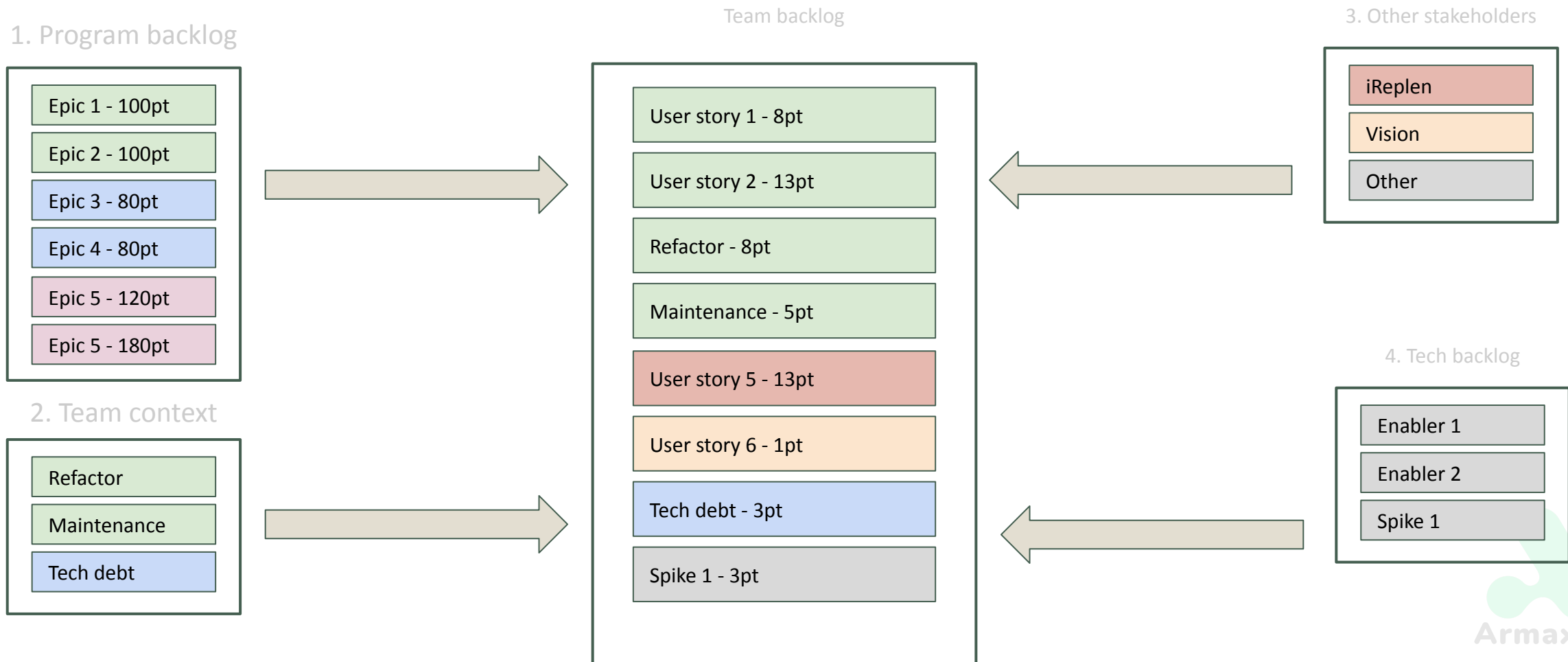
Execute 3 sprints (6 weeks)

Sprints

Value

Backlog

Backlog
User story 1 - 8pt
User story 2 - 13pt
Refactor - 8pt
Maintenance - 5pt
User story 5 - 13pt
User story 6 - 1pt
Tech debt - 3pt
Spike 1 - 3pt

Sprint 1
- User story 1 - 8pt
- User story 5 - 13pt

Sprint 2
- User story 2 - 13pt
- Refactor - 8pt

Sprint 3
- Maintenance - 5pt
- User story 6 - 1pt
- Tech debt - 3pt
- Spike 1 - 3pt

Epic 1 - 100pt — Happy path
Other stakeholders — Completed

Epic 1 - 100pt — Un-happy path
Team context — Completed

Team context — Completed
Other stakeholders — Completed
Team context — Completed
Tech backlog — Completed

Armaxon

USER STORY

A short description of functionality **told from the perspective of a user** that are valuable to either a user of the software or the customer of the software.

They typically follow a simple template:

As a **<type of user>**, I want **<some goal>** so that **<some reason>**.

User stories are often written on index cards or sticky notes, stored in a shoe box, and arranged on walls or tables to facilitate planning and discussion. As such, they strongly shift the focus from writing about features to discussing them.

In fact, **these discussions are more important than whatever text is written.**

Card
(User Story)

Anyone can write a US.
PO is responsible overall.

Conversation
(Discussed)

It's a joint effort to achieve everyone's understanding and agreement on the scope.

Confirmation
(Agreed)

Everyone understands WHAT needs to be done.

Hey... how about some examples?

As a power user, **I can** specify files or folders to backup based on file size, date created and date modified

As a user, **I can** indicate folders not to backup **So that** my backup drive isn't filled up with things I don't need saved

As a daughter whose mother has dementia, **I want** to know if she sleeps at night **AND** receive alerts when she doesn't

Armaxon

Why User Stories?

1

Something significant (if not magical) happens when requirements are put in the first person. Obviously by saying "As a such-and-such, I want ..." you can see how the person's mind goes instantly to imagining he or she is a such-and-such.

2

Having a structure to the stories actually helps the product owner prioritize. If the product backlog is a jumble of things like:

Fix exception handing | Let users make reservations | Users want to see photos | Show room size options

... and so on, the product owner has to work harder to understand **what the feature is**, **who benefits from it**, and **what the value** of it is.

3

I've heard an argument that writing stories with this template actually suppresses the information content of the story because there is so much boilerplate in the text.
"I apologize for such a long letter - I didn't have time to write a short one."
— **Mark Twain**

Armaxon

Here are few reasons to use story points:
- Dates don't account for the non-project related work that inevitably creeps into our days: emails, meetings, and interviews that a team member may be involved in.
- Dates have an emotional attachment to them. Relative estimation removes the emotional attachment.
- Each team will estimate work on a slightly different scale, which means their velocity (measured in points) will naturally be different. This, in turn, makes it impossible to play politics using velocity as a weapon.
- Once you agree on the relative effort of each story point value, you can assign points quickly without much debate.
- Story points reward team members for solving problems based on difficulty, not time spent. This keeps team members focused on shipping value, not spending time.

Definition Of Ready

- User Story has the following :
    - Name
    - Description
    - ACs should be written in BDD syntax
    - Story meets INVEST criteria
    - Attached (wireframe, UI) where applicable

- The User Story has been presented to the Team. Any questions they have regarding the story have been resolved. The story and feature file has been updated to reflect any new understanding.

- For any front end facing stories the UI design is sufficiently detailed and has been agreed within the team, PO is signed off the design.

- The story must be a child of Epic and fit within the requirements of that Epic

- All inter-team dependencies (where known) have been considered.

- The team have sized the story.

Definition Of Done

- All acceptance criteria are tested on UAT

- All acceptance criteria, design met, tested and approved by the Product Owner on UAT environment

- Delivery team have conducted exploratory testing on all agreed browsers and devices for any customer facing front end stories.
  Browsers are:
    - Chrome (latest version)
    - Firefox (latest version)
    - Internet Explorer 11
    - Default Android Browser (latest version)
    - Safari
    - If additional browser was not specified on story refinement session.

- Jira is updated with the relevant status

You can think of the Definition of Done as an extra set of acceptance criteria that are rubber stamped onto each and every user story.

Armaxon

WSJF = Cost of Delay / Job Size

CoD = User-Business Value + Time Criticality + Risk Reduction-Opportunity Enablement Value

Feature	User-Business Value	Time Criticality	RR-OE Value	Job Size	WSJF
Feature 1	13	13	8	20	**1.7**
Feature 2	3	8	13	5	**13.6**
Feature 3	8	5	5	8	**2.25**

The job with the highest WSJF is the next most important item to do.