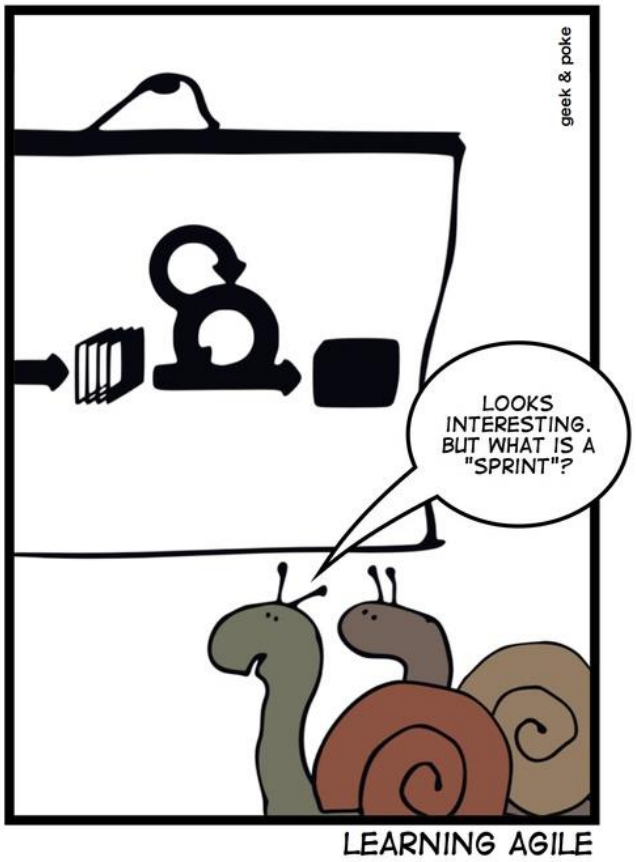


Comp 3120/8110

Software Development Management
Week 6 Lecture 2 –
Writing User Stories in detail

Definition of Ready
Definition of Done

Midterm Quiz 'hints and tips'

Log into Socrative!
http://www.socrative.com CALDWELL8573

Planning and the Project Plan

Assignment #2 is due at 5pm Friday Week 7 after the teaching break. Please work on this in your groups independently. There are no workshops between now and the due date for Assignment #2.

However, we will be monitoring Piazza to help with any questions you may have.

Where we left off Tuesday…

How to break down epics into stories

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

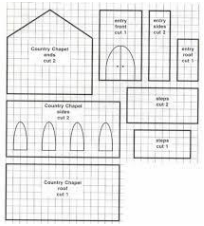
Often splitting stories **separate high-value parts from low-value parts**, allowing the team time to focus on the valuable parts of a feature. This helps to provide value by enabling development of a minimal, end-to-end solution and then filling in the rest of the solution. Frequently splitting stories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

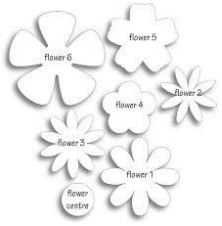
Decomposition takes practice and experience. Some stories are more difficult than others to split. Discovery conversation is one of the best tools for breaking down big stories.

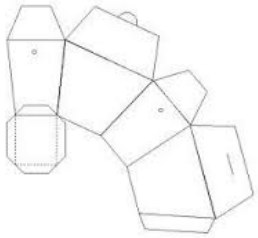
Remember to only decompose epics only when more detailed estimation is required

Flowchart for Splitting Epics

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







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.

User Stories in Detail:

Discovery

User Stories in detail - Discovery

A **user story's journey** begins as an idea – for a **new feature** or a **whole new product**!

The **conversation** starts out at a **high level**: a who-what-why discussion.

At this point in the discussion, a **decision to go ahead** with further investigation is made; is this a **feature/product** that is **worth spending time** on? This is called discovery.

Discovery isn't about building software. It's about building a **deeper understanding** of **what we could build**.

User Stories in detail - Discovery

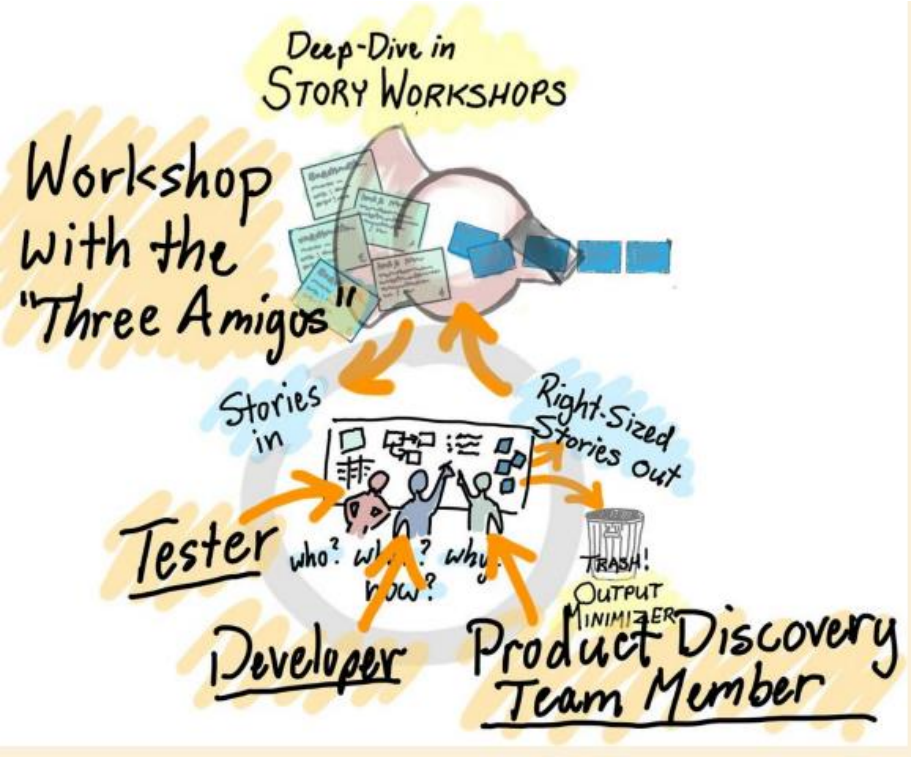
Discovery is about **asking** and **answering questions** such as:

- What *problems* are we really solving?
- What solutions could be *valuable* to the organisation and to customers buying or adopting the product?
- What does a *usable* solution look like?
- What's *feasible* to build given the time and tools that we have?

User Stories in detail – Backlog Grooming & Story-Writing Workshops

Story-writing workshops may happen every day, or all at once during a planning session. Frequently they happen during backlog grooming sessions.

Use story-writing workshops to discuss the details, to decompose stories and to really agree on exactly what you'll build.



Why would you include testers in a User Story Workshop?

User Stories in detail – Backlog Grooming & Story-Writing Workshops

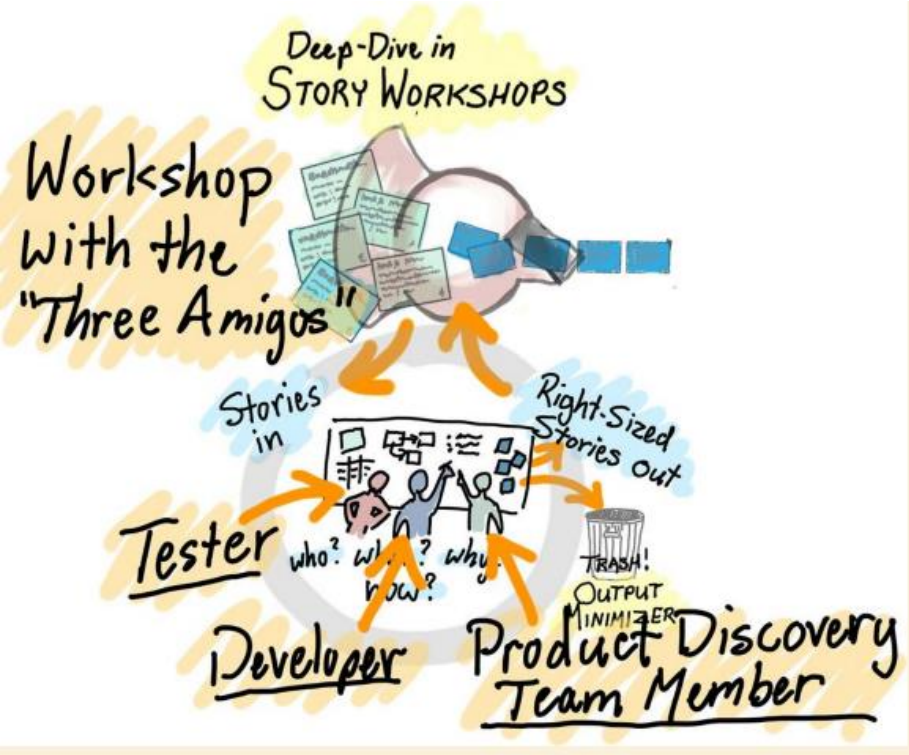
To develop quality, "ready" stories, **in addition to the usual product owner** you should aim to include a **tester**, a **member of the core product discovery team** and a **user experience designer** in your story-writing workshops.

Works through details of story

Agrees specific acceptance criteria

Conversation produces best estimate of how long story will take

Story probably then decomposed into ‘right-sized’ stories that can be built and tested in a few days



User Stories in detail – Backlog Grooming & Story-Writing Workshops

Start by framing the problem: why you're building something and who it's for.

Next, understand the customers and users so you gain a deeper insight into who they are and how the features will provide benefit for them.

To do this use simple personas. This helps look at the software through the eyes of the users.

A persona is an example of your target user assembled from the facts (and sometimes assumptions) you have about the users.

User Stories in detail – Backlog Grooming & Story-Writing Workshops

Now imagine the future!

Envision how the proposed solution will benefit your target users and customers and how they will use it.

Visualise the user interface to build shared understanding of the solution.

Combine ideas and refine them and come to a shared understanding of what the software could look like.

Don't forget to
"play 'what-about'?"

User Stories in detail – Backlog Grooming & Story-Writing Workshops

Don't forget to consider business rules especially important and complex rules, complex data validation, back-end services or systems that the product needs to connect with.

Finally, validate completeness and any technical concerns there may be. At this point, you should consider whether you have included too much. Are there bells and whistles here?

When workshopping stories the group should be small enough to work in front of a whiteboard,

i.e. **3-5 people** - a developer, a tester, and someone (s) who understands users and how the UI will look and behave.

User Stories in detail – In conclusion

Remember the client, product owner or whoever is in that role, understands the problem much better than they are able to predict how to solve it and the person who understands the technology is often the most qualified to solve the problem.

The client / developer relationship you're seeking is one like the good patient-doctor one where you tell the doctor what hurts and the doctor works out the best way to treat the problem as it is discovered.

User Stories in detail - recipe

Story-Writing Workshop Recipe (1 of 3)

Prior to the workshop make sure **everyone knows what stories** you'll be workshopping

Keep the workshop small so you remain productive

Include the right people -- someone who understands the users and how the UI should work (**product owner**, **UX professional** **or** **BA**), one or two **developers** who understand the codebase as they'll understand best if something is feasible; and a **tester** because they'll help ask the "what about" questions. There may be some others you want to include, but keep the group small. Sometimes people can wear two hats, for example a BA-tester.

User Stories in detail - recipe

Story-Writing Workshop Recipe (2 of 3)

For each story **dive deep and consider options**. You need to determine exactly

- who the user is
- how we believe it will be used
- what it looks like - that is, the user interface
- how the software behaves underneath the UI - i.e. the business rules and data validation stuff
- roughly how we might build the software (because we need to estimate/predict how long it will take to build

Agree on what to build so that you can answer

- What we will check to confirm this software is done
- How we will demonstrate this software later when we review it together

User Stories in detail - recipe

Story-Writing Workshop Recipe (3 of 3)

Talk and document - use whiteboards or paper to draw pictures, write examples and consider options. Record everything - don't let ideas vapourise. Take photographs, transcribe notes and drawings later.

Speak in examples - try and use specific examples of what users do, exactly what data might be entered, exactly what users would see in response, or whatever examples best support your story

Split and thin -- when discussing detail and development time, you'll often find stories are larger than should be put into your development cycle. Work together to split up the big stories and to "thin" them be removing unnecessary parts.

User Stories in detail - recipe

Story-Writing Workshop Caveats

Watch out for derailments:

Your workshop is not working when:

no-one participates - when one person describes and the rest listen

the focus is on acceptance criteria and not on telling the story about who does what and why

the options aren't considered from both a functional and technical perspective

User Stories in detail – In conclusion

Your job is to take the vision of the business stakeholder and help them make it a success.

Sometimes this may mean telling those stakeholders things they don't want to hear.

Working with stories is a continuous process of conversation and discussion to break them down from big things to small things.

During these conversations the **focus must be kept on what is being built, for whom and why**.
No matter how "ready" your stories appear, the conversation needs to keep going even when you're building because no matter how good they are, you won't have predicted everything you'll learn once you start to build.

Getting to Ready

Agile planning takes a **"just-in-time"** approach to developing details.

As a result, user activities (software features) are captured in a largely imprecise manner.

Only enough detail is known about them at any point during the development process to allow the tasks to be completed at that time.

But what about when it is time to develop that specific user activity (software feature)?

Well, then it is time to

get to "ready"!

Graphic from: https://www.youtube.com/watch?v=a911H-86qgc

In Agile Project Management what does ready mean?

What does it mean to be “ready”?

Feature(s) to be developed in sprint iteration need to be:

- Immediately actionable by the team
- Product backlog is designed to be a negotiation between the product owner and the team – so talk it through before planning the sprint
- Needs to add value – avoid ‘junk stories’
- Needs to be able to be estimated
- Needs to be testable
- Needs to be sized right for the team

See ‘Scrum – What does it mean to be Ready Ready’ | Graphic by https://www.savv-e.com.au

What else does it mean to be ready”?

A story that is "ready”:

- is defined clearly enough that all members of the team understand what must be done
- includes a clear statement of resulting business value that allows the product owner to priortise
- includes any required enabling specifications, wireframes etc

A story is **not "ready" if it depends on something outside the team’s control**.

Such stories **greatly increase the risk of an iteration failing** to achieve its goal, and you can’t do anything about it!

What else does it mean to be ready”?

A story that is "ready”:

- is estimated and sized such that it can be completed easily within a single iteration and
- is free from external dependencies, i.e. there is nothing beyond the team's control that must be done first in order to complete the story
- fully meets the INVEST (Independent, Negotiable, Valuable, Estimable, Small and Testable) criteria for user stories

A story is **not "ready" if it depends on something outside the team’s control**. Such stories **greatly increase the risk of an iteration failing** to achieve its goal, and you can’t do anything about it!

What makes a GREAT User Story?

A good model to follow Bill Wake's INVEST acrynym. He suggests User Stories should be:

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

APM Planning and Introduction to User Story Decomposition

Project roadmap and release planning produce a collection of coarse-grained stories loosely mapped to releases when the product backlog – the User Story Map (USM) was created.

At the time of the sprint the stories at the top of the product backlog / User Story Map need to be ready to be pulled into the sprint backlog. What does this mean?

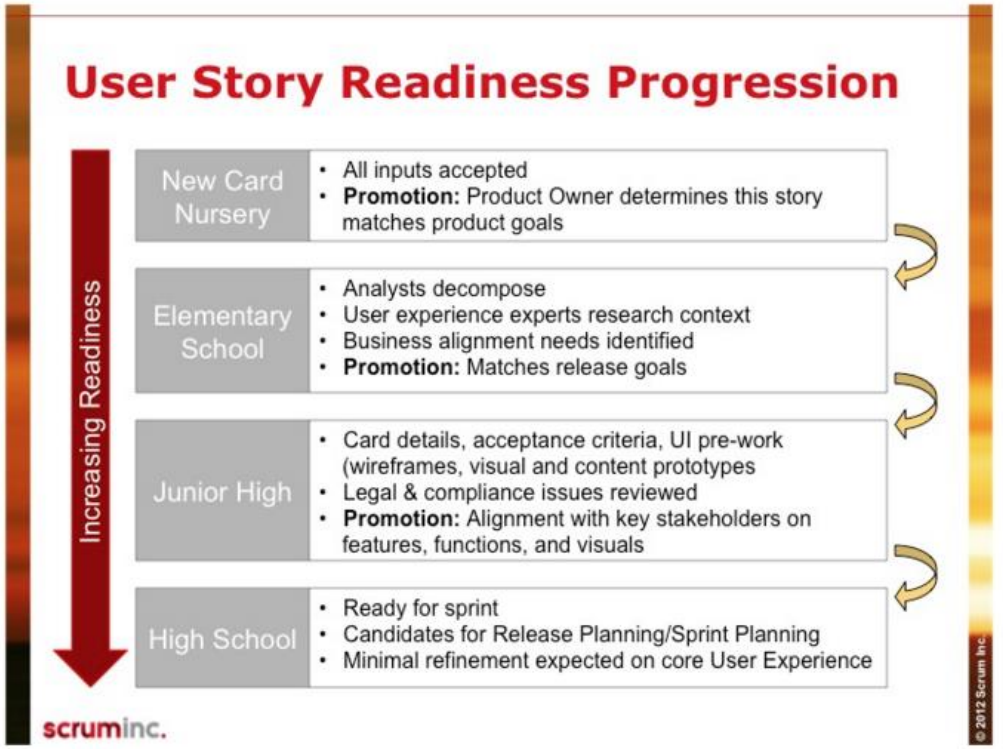
The story needs to be ready to go from abstract to concrete.

Moving towards "ready“

The product owner plays an important role in supporting the team to clarify stories. As part of "grooming" the product backlog the team regularly estimates stories.

This takes about 10% of team time and is often carried out one day each week with an update during the iteration planning meeting.

The step from "estimable" to "estimated" in terms of duration may be undertaken during the iteration planning meeting.



Why is it important that a team only include "ready" stories in the sprint backlog?

APM Planning and Introduction to User Story Decomposition

Having a definition of “ready” and sticking to it…

…helps a team avoid wasted effort by working on features…

…that do not have clearly defined criteria for “done”.

http://lifehacker.com/5957613/keep-your-task-lists-simple-with-columns-for-ready-doing-and-done

Ready and Done

Getting to Done

What does the term "done" mean in Agile Project Management?

What does it mean to get to “done”?

- What is the definition of “done”?
- Conditions of satisfaction
- What does it mean to be “done”?
- Benefits of “done”
- Using “done”

"Done" refers to a set of special conditions of satisfaction applying to every item on the product backlog.

Development 'done-ness'

Inspected code

Checked in code

Well-written code

Automated tests included

Documentation complete

Acceptance criteria 'done-ness'

E.g. User story: "As a user, I am required to login before using the site" :

User can only login with proper credentials

Includes a "remember me" option

User can request a password reminder

User is locked out after 3 failed attempts

Definition of “done” DoD

The **DoD may vary** depending on whether it relates to a **story**, an **iteration**, a **release** or **the whole product**.

When applied to a story, "done" means:

the feature has been developed AND

tested AND

meets all required acceptance tests AND

anything else that is included in the DoD for that particular story.

Definition of “done” DoD

Ideally it means the **story could be shipped to the customer**.

A team’s definition of done is an agreed-upon set of things that must be true before any product backlog item is considered complete.

In some ways, **"done" is like a rubber stamp** applied after it has been independently verified as meeting all the requirements of "done".

What are the benefits of using the concept of "done" when using agile software development?

Benefits of "done"

Establishing “done” up front can save hours of:
refactoring,
- mis-work,
- busy-work,
- miscommunication,
- hidden work.

‘Done’ is a commitment to quality - provides a means to communicate - everyone understands what the answer means!

Using "done"

Don’t demonstrate stories in the Iteration Meeting that don’t meet DoD

“Done” is about product quality, **not approval from external parties**

If teams don’t get many stories “done” in their iterations it may indicate:

- poor story writing,
- too large stories or
- too many external dependencies

Multiple teams working on the same product will need a **shared DoD**

What happens when you demonstrate stories that aren't yet "done"?

User Stories in detail – In conclusion

"***If all this is starting to make sense***, ***then you've made that big, necessary mindshift***. It's not a shift to use stories to document requirements, but a shift toward working with people more effectively, and together focusing on solving real problems with the products you create."

Australian National University

10 minute break

Quiz preparation

- **Quiz structure**
- **Sample questions & approach to marking**

https://www.telegraph.co.uk/education/educationopinionl

Online quiz provisions

Your quiz will be delivered via Wattle.

Wattle offers online approaches for all the types of questions that we will ask in the midterm quiz.

You may bring one A4 page of notes (printed on one side

We will not be using invigilation software.

I do however reserve the right to randomly or otherwise request meetings with students to (verbal questioning) confirm the understanding shown in the quiz.

Important to remember!

- We **MUST** be able to read your writing (in this case – typing)
- Your English expression should be:
  - Coherent, fluent and appropriate for academic purpose, as well as grammatically correct
  - Appropriately and logically structured
- Your answers should be **concise**
- Build a defensible argument
- Stick to the facts - long, rambling answers are more likely to lose marks than to gain them

Quiz

- It is YOUR RESPONSIBILITY to confirm date and time via the official examination timetable and to identify the Wattle quiz and USM submission links in Wattle (which will be prominently displayed).
- 1 hour, written quiz with reading period beforehand.
- Permitted Materials: 1 A4 sheet of paper with written or printed notes.

The quiz will challenge you!

- To get more than a pass you MUST demonstrate higher order thinking skills – you need to be able to **analyse**, **evaluate**, **create** new ideas, and communicate those ideas in writing

Analyse: Separate the whole into its component parts

Evaluate: Develop opinions, judgements or decisions

Create/Synthesise: Combine ideas to form a new whole

Quiz structure

- **Project Management Knowledge**
  - Consisting of several short answer, multiple choice and true/false/explain questions worth between 1 and 5 marks each
  - Answer **ALL** questions

The difference between
Comp3120 and Comp8110 exams

Some questions will be different.

and

We will hold Comp8110 students to a higher standard

Image: https://www.notsowimpyteacher.com/2016/09/5-hands-on-ways-to-teach-multiplication.html

Project Management Knowledge

- Several questions intended to show your broad understanding of the course
- Questions are self-explanatory
- Answer ALL questions – even the smallest response may get you ½-1 mark! Don't waste it. Seriously.

Overall timing and marking

- You should use the size allowed for the answer space and the number of marks attached to the question as a guide to the depth of the answer expected.
- *Very approximately*, one short sentence or phrase for each 0.5 mark is expected
- **There will not be enough time to be getting all your answers out of your notes**.
- Be careful of overuse of your notes. Irrelevant statements taken from your notes to pad answers will detract from your good answers.
- Quiz marking will be done by your tutors and I in tandem.

Project Management Knowledge

Q1(a) [2 marks] Not all software development is a project. Identify and describe two key characteristics of a project which distinguish it from ”business as usual”.

Project Management Knowledge

Q1(a) [2 marks] Not all software development is a project. Identify and describe two key characteristics of a project which distinguish it from ”business as usual”.

Need to **identify**, i.e. name two key characteristics

Need to **describe** each of the characteristics

Project Management Knowledge

Q1(a) [2 marks] Not all software development is a project. Identify and describe two key characteristics of a project which distinguish it from ”business as usual”.

- Finite – has a start and end date
- Unique – develops / delivers a unique product or service
- Uses budget of various (limited) resources – human and money
- Sponsored - Involves a customer / sponsor for the project and its goals
- Change – a project aims to create change to organisation

Project Management Knowledge

Q1(c) [2 marks] Agile means no documentation [true] [false]
Explanation:

Project Management Knowledge

Q1(c) [2 marks] Agile means no documentation [true] [false]

Explanation:

1 mark for getting true or false correct

1 mark for the explanation of your answer

Sometimes this means you may get the true / false wrong but you give a good explanation so get a single mark.

Project Management Knowledge

Q1(c) [2 marks] Agile means no documentation [true] [false]
Explanation:

- False
- Agile focus is on producing working software instead of spending large amounts of time creating documentation UP FRONT
- Effective agile delivers focused, value-driven, business-beneficial documentation enabling the business to use the product effectively & the technical team to support it.

Project Management Knowledge

Q1(e) [4 marks] Project management is frequently called a profession. Identify and describe two attributes of professional behaviour and using examples explain why they are important.

Project Management Knowledge

Q1(e) [4 marks] Project management is frequently called a profession. Identify and describe two attributes of professional behaviour and using examples explain why they are important.

- 1 mark for identifying, i.e. naming, two attributes – half a mark each
- 1 mark for describing – half a mark each description
- 2 marks for explanation – half a mark explaining each attribute

Project Management Knowledge

Q1(e) [4 marks] Project management is frequently called a profession. Identify and describe two attributes of professional behaviour and using examples explain why they are important.

- Respect -- Show respect for people around them regardless of role they occupy
- High emotional intelligence -- Exhibit high emotional intelligence by considering emotions and needs of others,
- Respect/rational behaviour -- Don't let a bad day impact how they interact with colleagues and clients,
- Accept responsibility/commitment -- Make and keep commitments,
- Life long learning -- Continually learn through reflection
- Good time-management skills,
- Trustworthy / honest / ethical

Review

Key topics

All material in the course so far – review lectures, workshops, and pre-reading