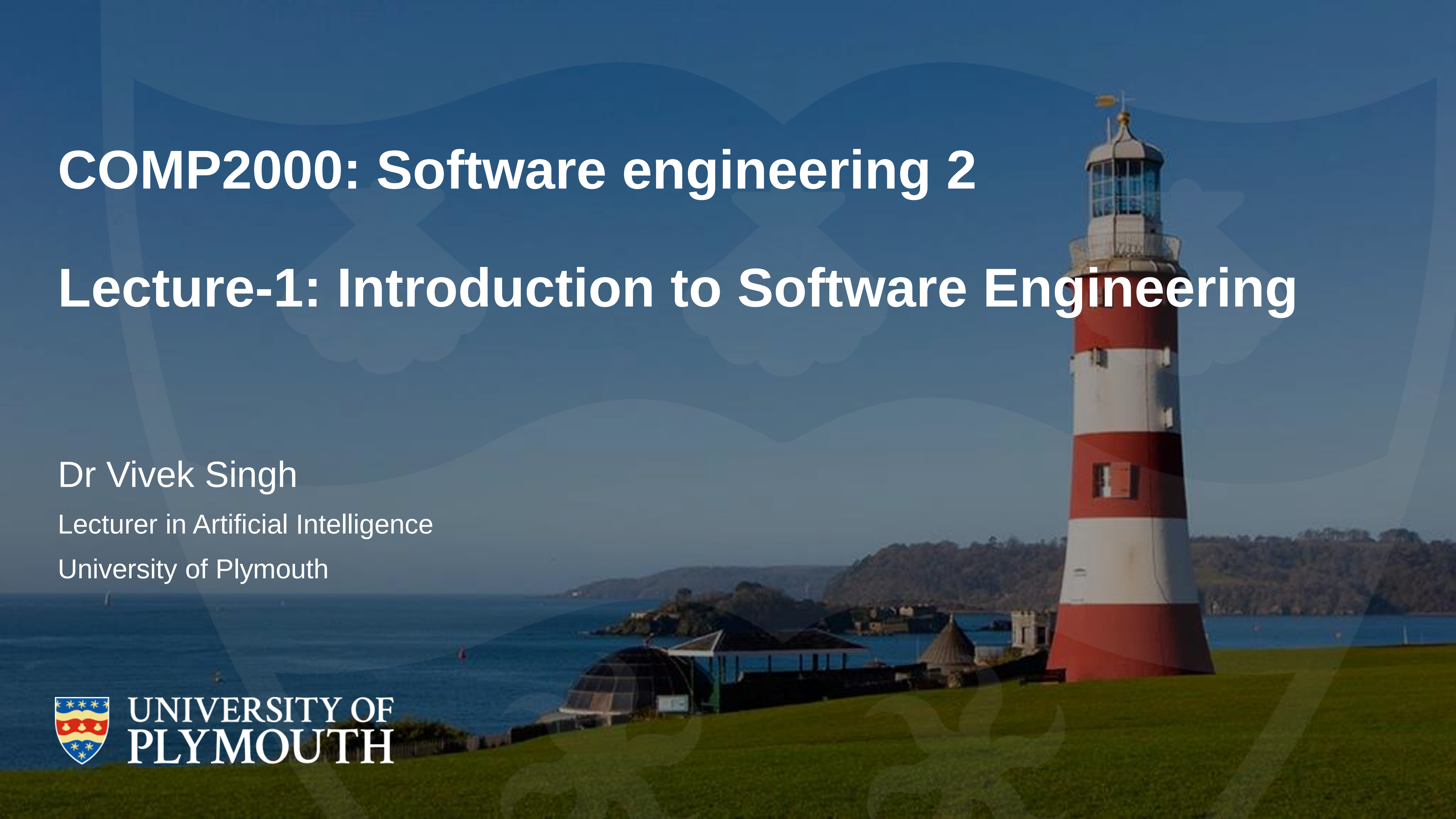


COMP2000: Software engineering 2

Lecture-1: Introduction to Software Engineering

Dr Vivek Singh

Lecturer in Artificial Intelligence

University of Plymouth

UNIVERSITY OF PLYMOUTH

About team

Dr Vivek Singh

Lecturer in Artificial intelligence

Office: PSQ A317

Email: vivek.singh@plymouth.ac.uk

Research area: Computationally efficient learning systems, Multimodal learning, Multi-task learning, Surgical action analysis, Meta learning, etc.

Dr Vivek Singh

Lecturer in Computer Science, School of Engineering, Computing and Mathematics

https://orcid.org/0000-0003-1728-1198

Contact expert

View Scopus Profile

Accepting PhD Students

Full team: Dr Vivek Singh (Module lead)

Dr Ali Al-Nuaimi and Dr Vassilis Cutsuridis (Labs)

UNIVERSITY OF PLYMOUTH

Course objective

- To learn about topics that instil best practice into the students' software development activity.
- To explore a range of commonly used programming paradigms in mobile development.
- To understand the benefit of using standardised design patterns.

Assessments

- Coursework

• Set exercises	30%
• Report	70%

- The coursework will focus on develop of an android application using java.
- Need to use all the concepts that we will learn in the lectures.

UNIVERSITY OF PLYMOUTH

Outline

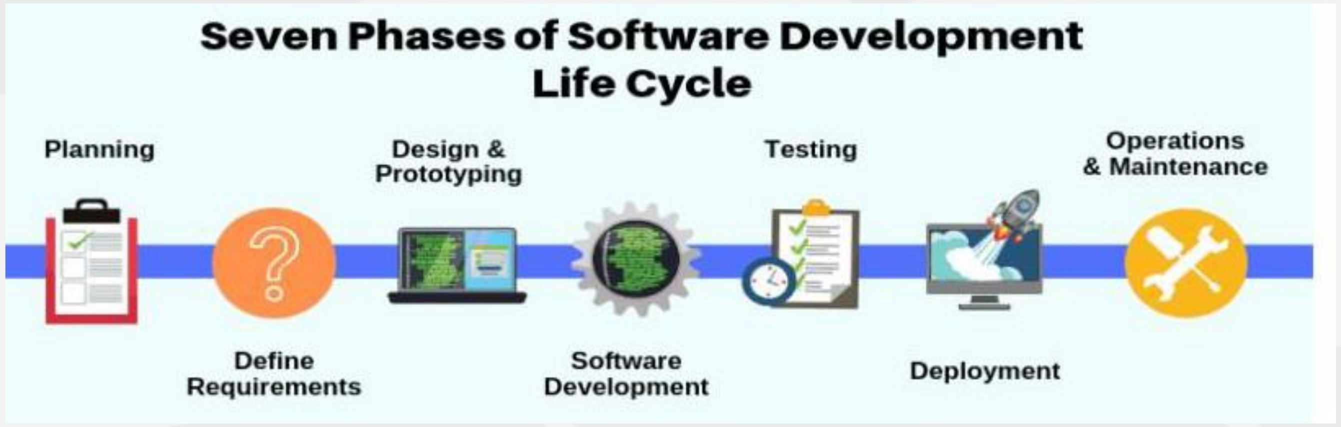
- Overview of software engineering
- Introduction to design process
- Software development models
- Prototyping
- Conceptual Design
- Concrete Design
- Using Scenarios
- Construction

What is Software Engineering?

- Software engineering is defined as a process of:
  - analysing user requirements and then designing, building, and testing software application which will satisfy those requirements.
- The process of building a software (software development) involves a series of steps from start to end.
- This is called Software Development Life Cycle (SDLC)

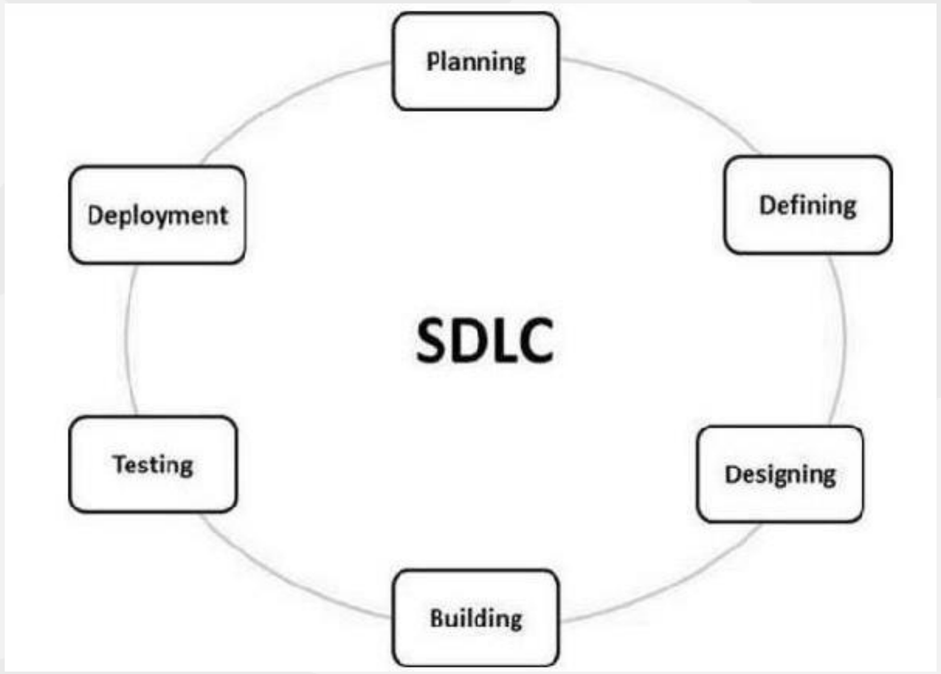
SDLC

- SDLC: is the application of standard business practices to building software applications.
- It’s typically divided into seven steps:



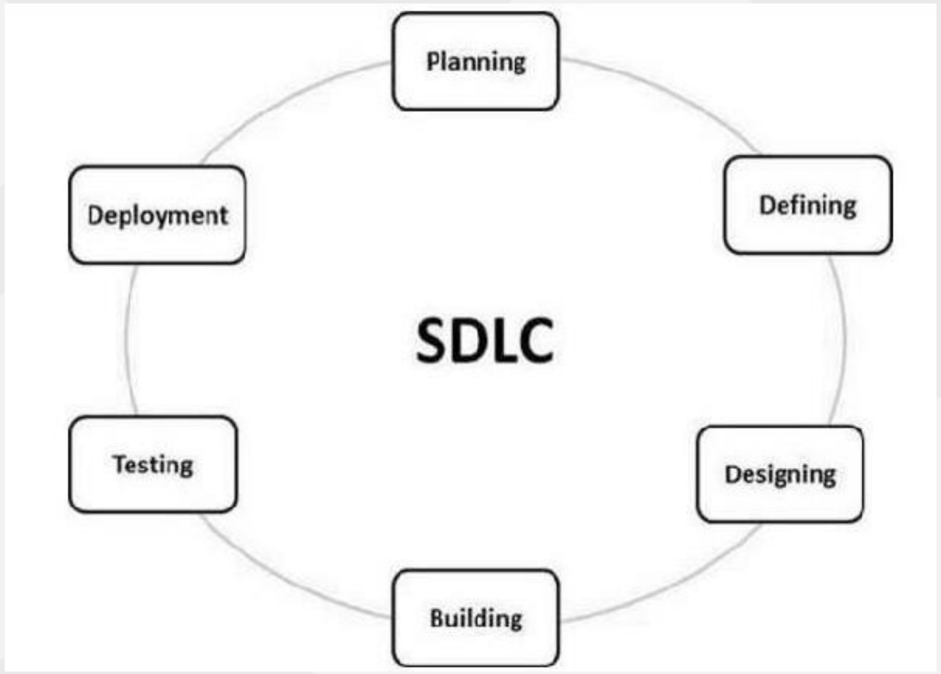
SDLC

- Planning and Requirement Analysis
- Defining Requirements
- Designing the Product Architecture
- Building or Developing the Product
- Testing the Product
- Deployment in the Market and Maintenance



SDLC

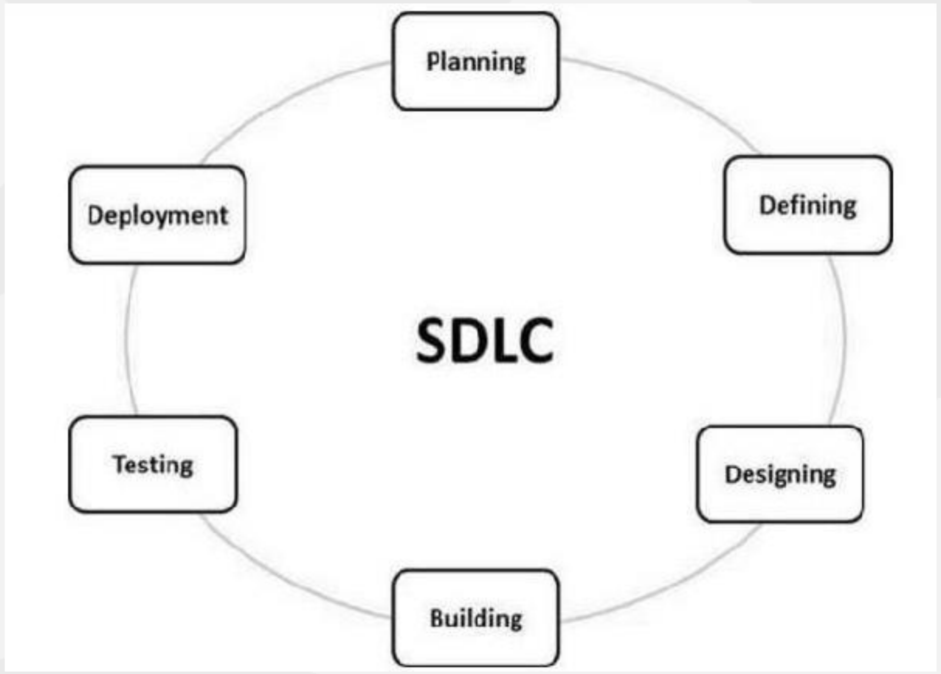
- Planning and Requirement Analysis
    - Feasibility analysis
    - Risk management planning
    - Budget estimation and schedule planning
- Defining Requirements
- Designing the Product Architecture
- Building or Developing the Product
- Testing the Product
- Deployment in the Market and Maintenance



UNIVERSITY OF PLYMOUTH

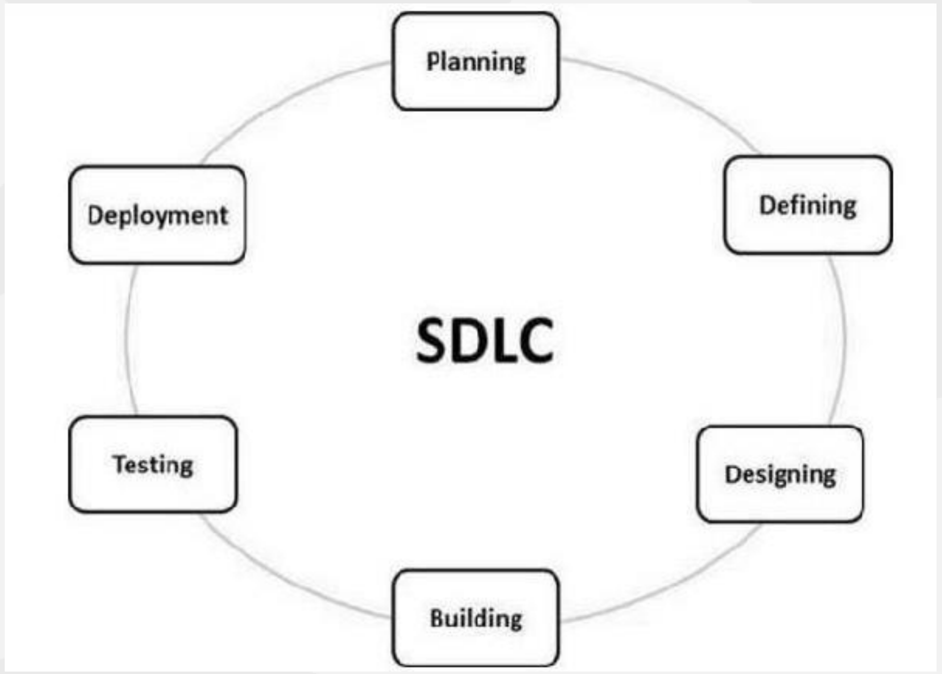
SDLC

- Planning and Requirement Analysis
- Defining Requirements
    - Understand the functional and non-functional requirements
    - Engage stakeholders to gather requirements
    - Define use cases, user stories, or system specifications.
- Designing the Product Architecture
- Building or Developing the Product
- Testing the Product
- Deployment in the Market and Maintenance



SDLC

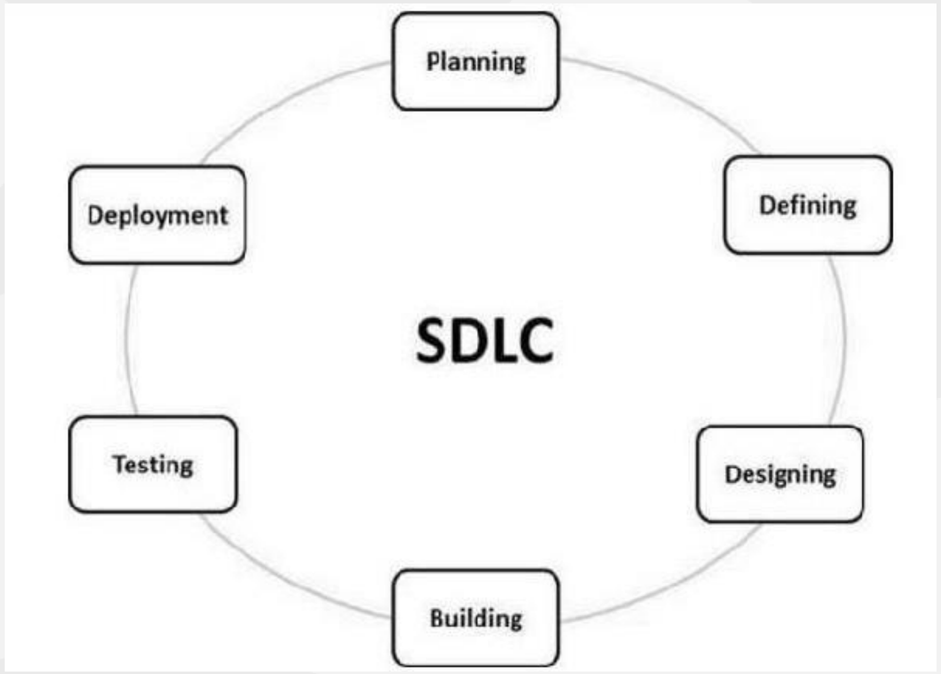
- Planning and Requirement Analysis
- Defining Requirements
- Designing the Product Architecture
  - blueprint for the system architecture and design elements
  - High-level design (HLD)
  - Low-level design (LLD)
- Building or Developing the Product
- Testing the Product
- Deployment in the Market and Maintenance



UNIVERSITY OF PLYMOUTH

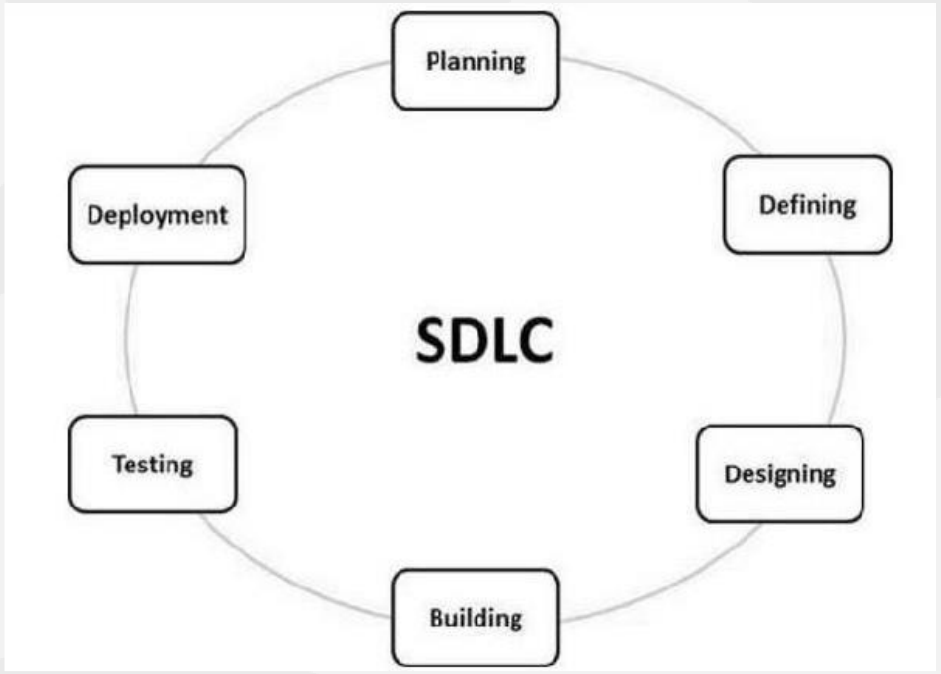
SDLC

- Planning and Requirement Analysis
- Defining Requirements
- Designing the Product Architecture
- Building or Developing the Product
  - Developers write code according to the design documents
  - Unit testing to ensure correct working of components.
- Testing the Product
- Deployment in the Market and Maintenance



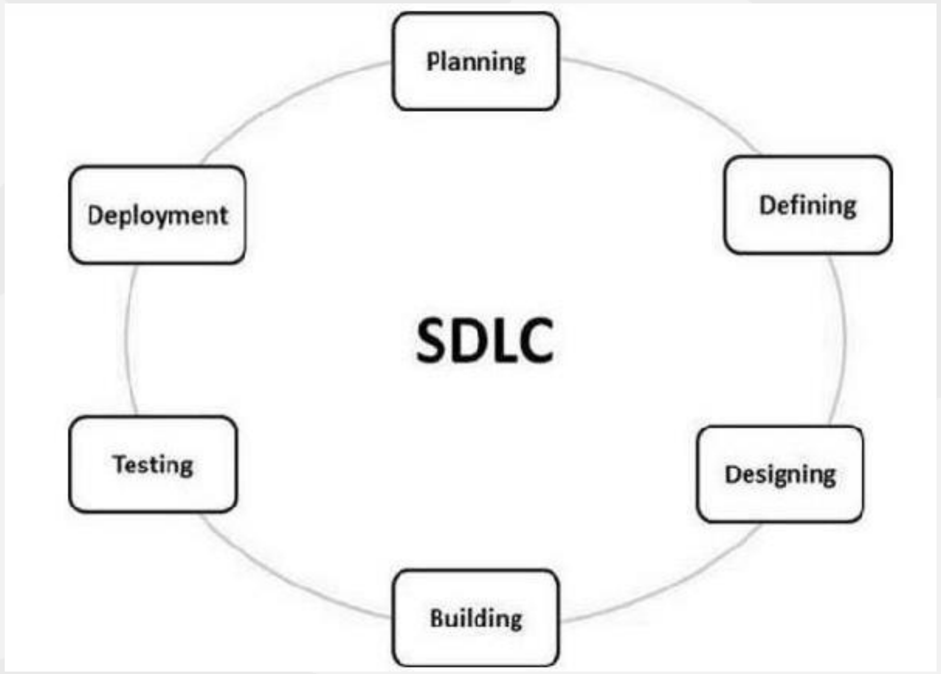
SDLC

- Planning and Requirement Analysis
- Defining Requirements
- Designing the Product Architecture
- Building or Developing the Product
- Testing the Product
    - Functional, integration, system, and acceptance testing.
    - Identify and fix defects
- Deployment in the Market and Maintenance



SDLC

- Planning and Requirement Analysis
- Defining Requirements
- Designing the Product Architecture
- Building or Developing the Product
- Testing the Product
- Deployment in the Market and Maintenance
    - Deploy software to production environments.
    - Provide documentation, training, and support
    - Fix issues or bugs discovered post-deployment.
    - Perform enhancements and upgrades.



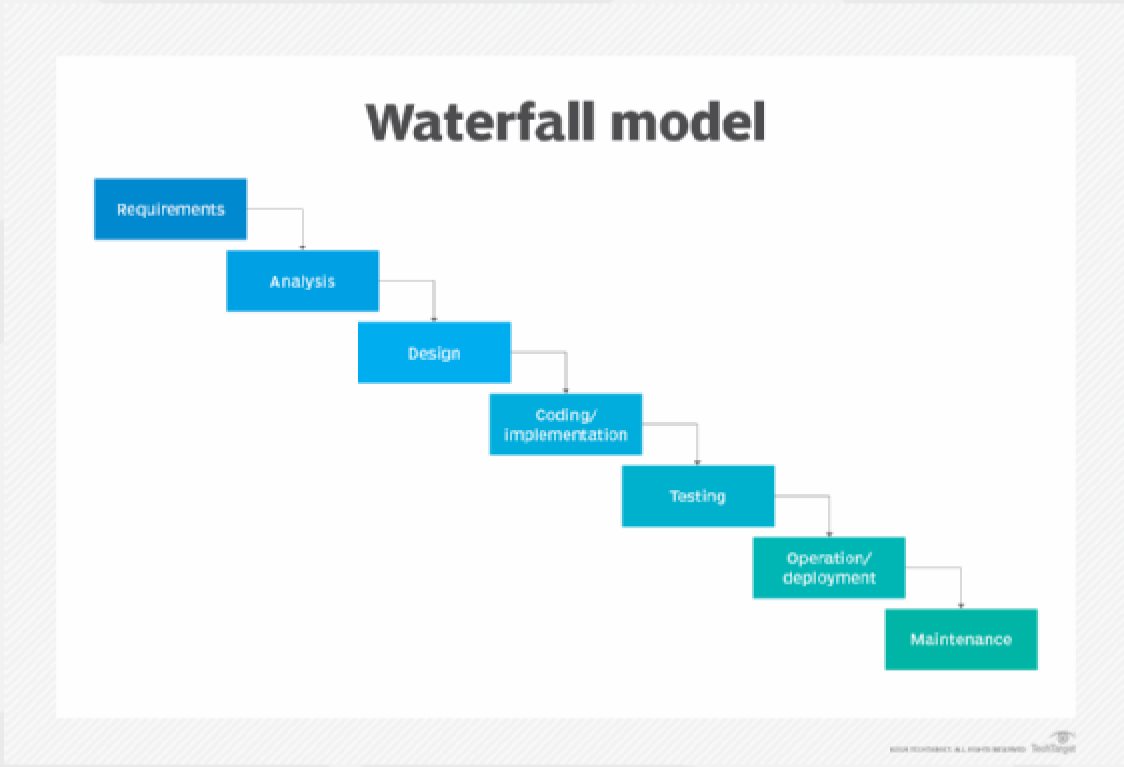
UNIVERSITY OF PLYMOUTH

SDLC models

- Waterfall model
- Agile model

Waterfall model

- The waterfall model arranges all the phases sequentially so that each new phase depends on the outcome of the previous phase.
- Conceptually, the design flows from one phase down to the next, like that of a waterfall.
- Useful in small projects or in safety critical applications.



https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model

Agile model

- It refers to a software development approach based on iterative development.
- Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning.
- Each iteration is considered as a short time "frame" in the Agile process model (one to four weeks).
- It minimizes the project risk and reduces the overall project delivery time

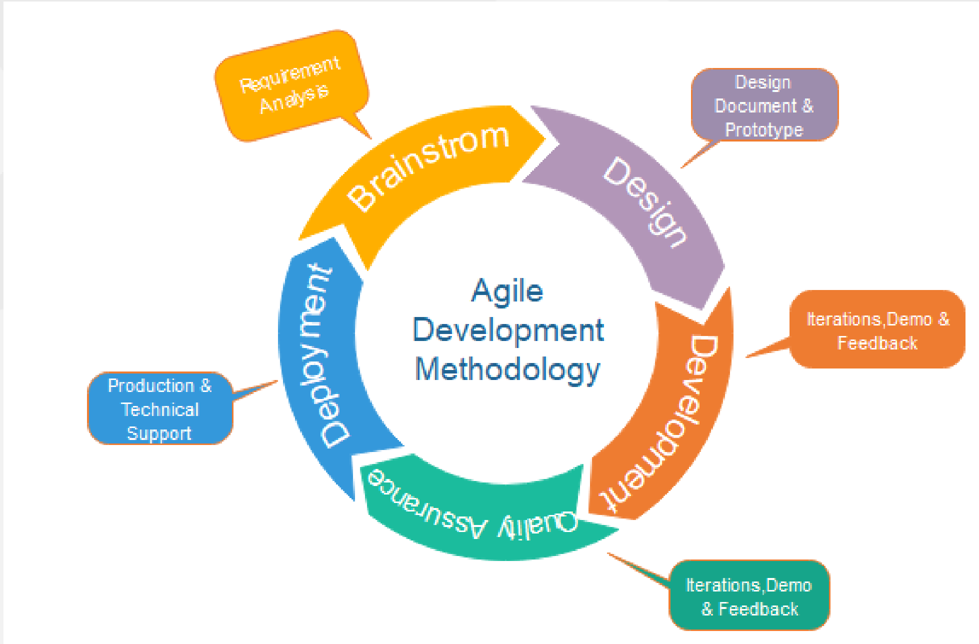

Requirement Analysis
Design Document & Prototype
Brainstrom
Design
Agile Development Methodology
Deployment
Development
Iterations,Demo & Feedback
Production & Technical Support
Quality Assurance
Iterations,Demo & Feedback

Fig. Agile Model

https://www.javatpoint.com/software-engineering-agile-model

UNIVERSITY OF PLYMOUTH

Design activities

- Design activities begin once some requirements have been established.
- The design emerges iteratively, through repeated design-evaluation-redesign
- cycles involving users.

UNIVERSITY OF PLYMOUTH

Design activities

- Design activities begin once some requirements have been established.
- The design emerges iteratively, through repeated design-evaluation-redesign
- cycles involving users.

The process of the design involves four basic activities:

- Establishing requirements
- Designing alternatives
- Prototyping
- Evaluating

- Mentimeter
- Go to: https://www.menti.com/
- Code: **4995 2474**

UNIVERSITY OF PLYMOUTH

- Mentimeter
- Go to: https://www.menti.com/
- Code: **4378 9111**

UNIVERSITY OF PLYMOUTH

Types of designs

- There are two types of design: conceptual and concrete.
  - Conceptual design is concerned with developing a conceptual model that captures what the product will do and how.
  - Concrete design is concerned with details of the design such as menu structures, haptic feedback, physical widgets, and graphics.
- As design cycles become shorter, the distinction between these two becomes blurred
- but they are worth distinguishing because each emphasizes a different set of design concerns.

Prototyping

- For users to evaluate the design of an interactive product effectively, designers must prototype their ideas.
- In the early stages of development, these prototypes may be made of paper and cardboard, or ready made components pulled together to allow evaluation, while as design progresses, they become more polished, compact, and robust so that they resemble the final product.
- The design process may start from two distinct situations: when starting from scratch or when modifying an existing product
- Much of design comes from the latter (i.e. modifying an existing product), and it is tempting to think that additional features can be added, or existing ones tweaked without extensive investigation, prototyping, or evaluation
- Although prototyping and evaluation activities can be reduced if changes are not significant, they are still valuable and should not be skipped
- In this lecture, we look at the activities involved in progressing a set of requirements through the cycles of prototyping to construction.

Prototyping

- It is often said that users can't tell you what they want
- but when they see something and get to use it, they definitely know what they don’t want.
- Having established some requirements, the next step is to try out design ideas through prototyping and evaluation cycles.

What is prototype

- A prototype is one manifestation of a design that allows stakeholders to interact with it and to explore its suitability;
- it is limited in that a prototype will usually emphasize one set of product characteristics and de-emphasize others.
- When you hear the term prototype, you may imagine a scale model of a building or a bridge, or a piece of software (that crashes every few minutes).
- A prototype can also be a paper based outline of a display, a collection of wires and ready made components, an electronic picture, a video simulation, a complex piece of software and hardware, or a three dimensional mockup of a workstation.

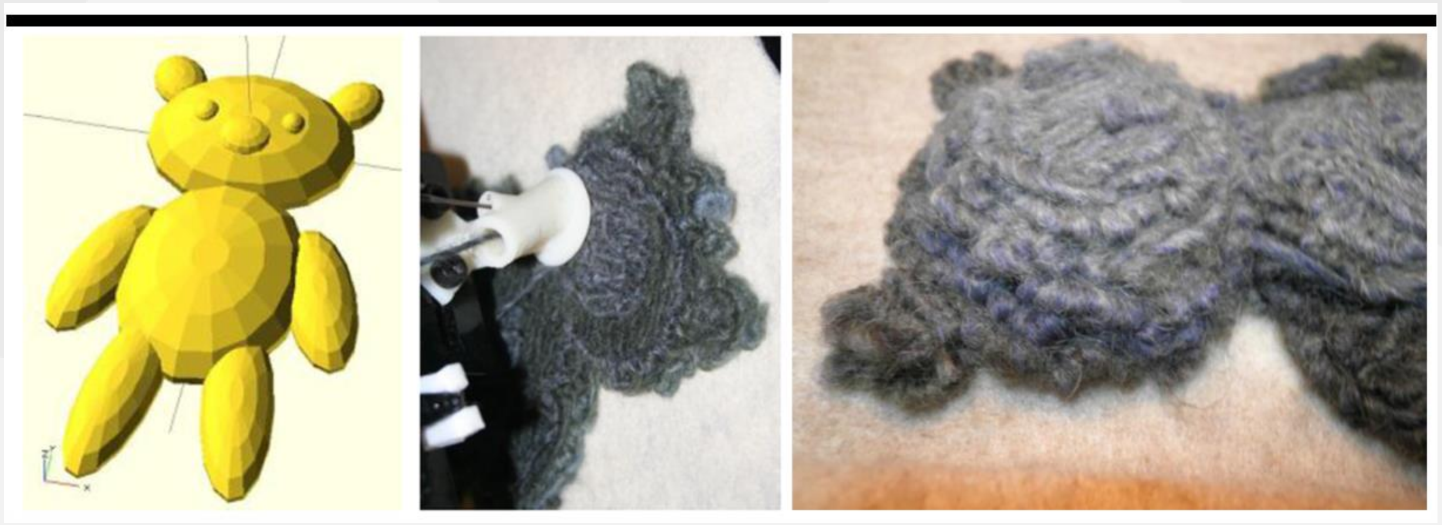


Figure: teddy bear ‘printed’ from a wireframe Design

Source: https://la.disneyresearch.com/publication/printing-teddy-bears-a-technique-for-3d-printing-of-soft-interactive-objects/

Why Prototype?

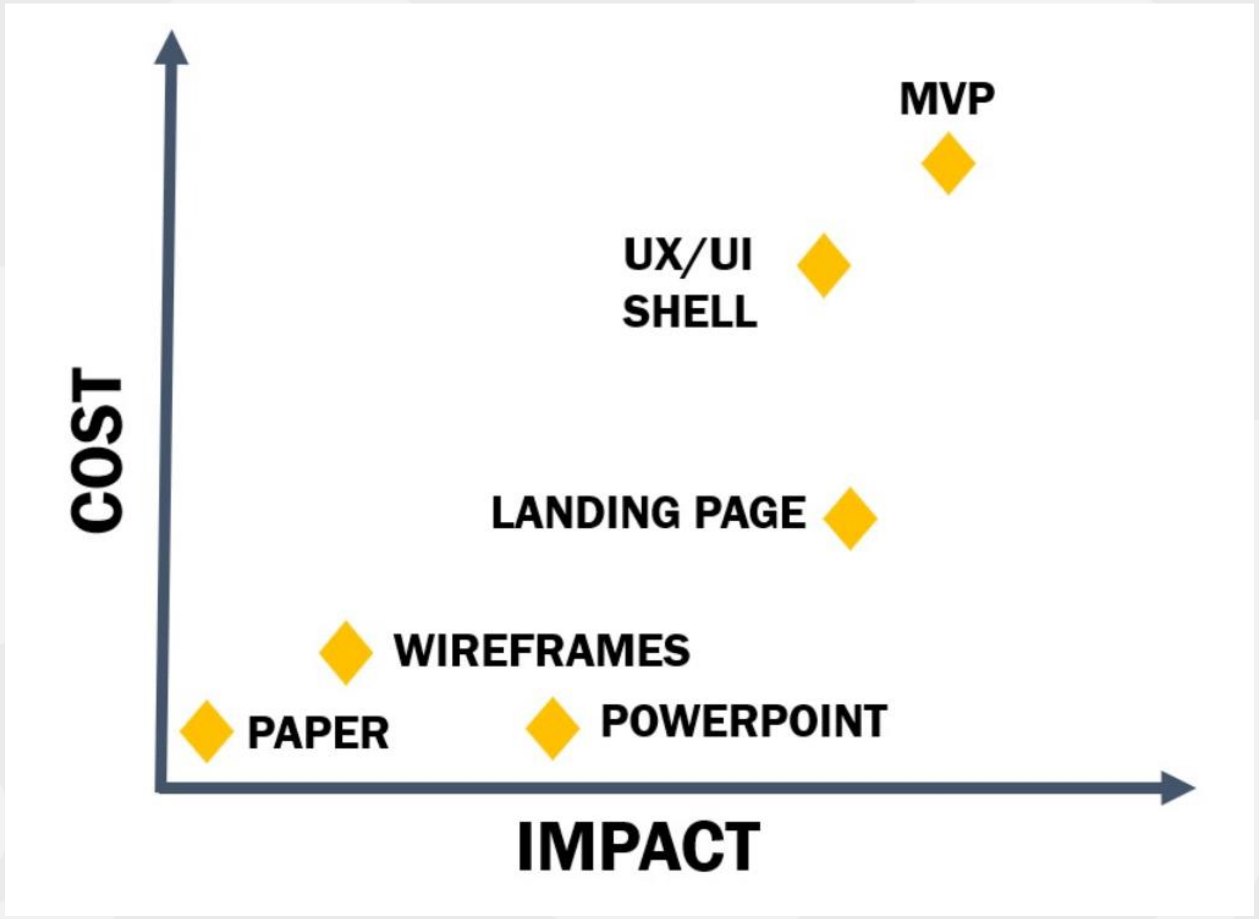
- Prototypes are useful when discussing or evaluating ideas with stakeholders
- They are a communication device among team members, and an effective way for designers to explore design ideas.
- The activity of building prototypes encourages reflection in design.
- Prototypes answer questions and support designers in choosing between alternatives.

  For example , to test out the technical feasibility of an idea , to clarify some vague requirements, to do some user testing and evaluation, or to check that a certain design direction is compatible with the rest of product development
- The purpose of your prototype will influence the kind of prototype you build.

- The prototype shows the intended functions and buttons, their positioning and labelling, and the overall shape of the device,
- but none of the buttons actually work.
- This kind of prototype is sufficient to investigate scenarios of use and to decide design elements
  - for example, whether the button images and labels are appropriate and the functions sufficient,
  - It does not test whether the speech is loud enough or the response fast enough.

Subjective Cost/Impact Scale

- Figure: relation between the impact and associated cost for different type of solutions



UNIVERSITY OF PLYMOUTH

Low Fidelity Prototyping

- A low fidelity prototype does not look very much like the final product and does not provide the same functionality.
- For example , it may use very different materials, such as paper and cardboard rather than electronic screens and metal,
- it may perform only a limited set of functions, or
- it may only represent the functions and not perform any of them.
- Low fidelity prototypes are useful because they tend to be simple, cheap, and quick to produce.

Low Fidelity Prototyping

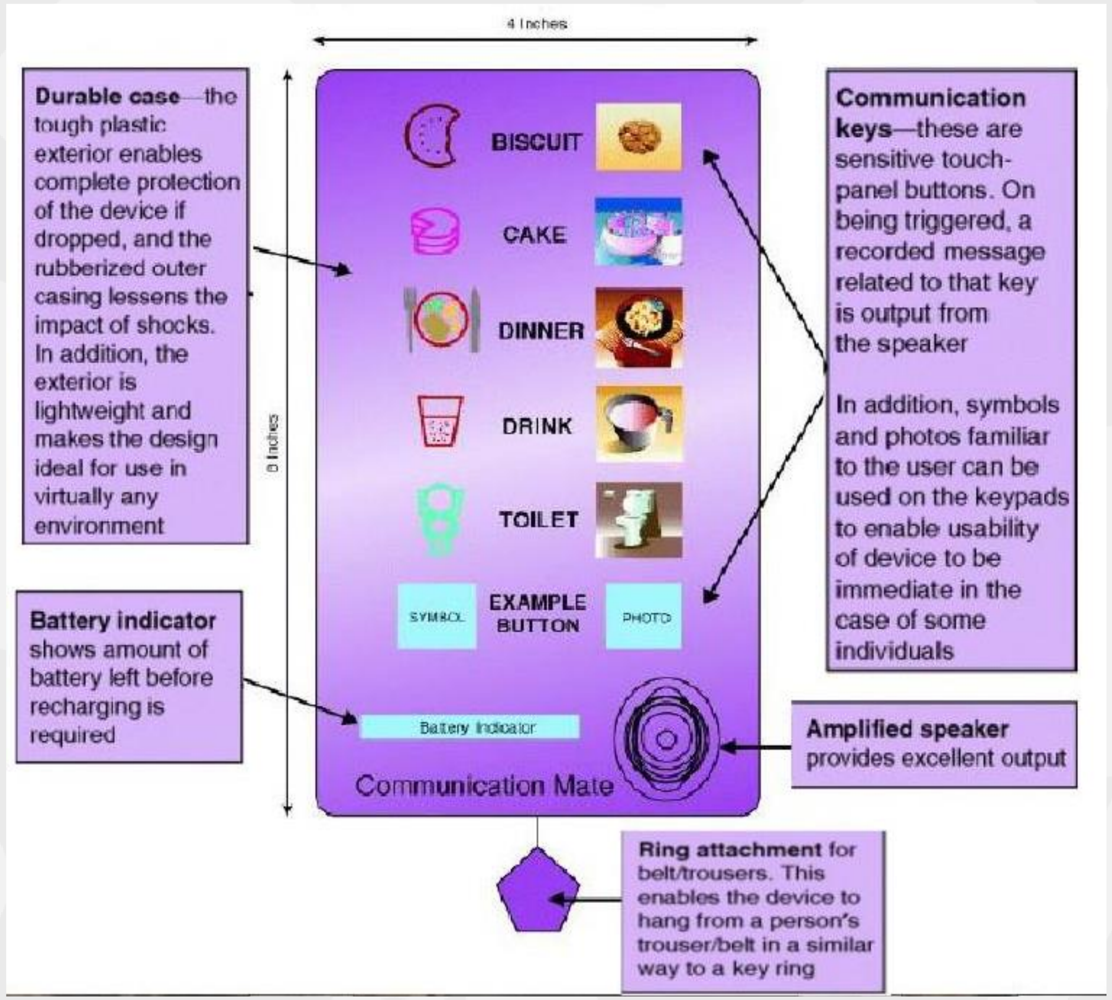
- This also means that they are simple, cheap, and quick to modify so they support the exploration of alternative designs and ideas.

Figure: Example of low fidelity vs high fidelity prototype

Low Fidelity Prototyping

- This is particularly important in early stages of development , during conceptual design
  - for example, because prototypes that are used for exploring ideas should be flexible and encourage exploration and modification.
- Low fidelity prototypes are not meant to be kept and integrated into the final product. They are for exploration only

Figure: A paper based prototype


4 Inches
8 Inches
Durable case—the tough plastic exterior enables complete protection of the device if dropped, and the rubberized outer casing lessens the impact of shocks. In addition, the exterior is lightweight and makes the design ideal for use in virtually any environment
BISCUIT
CAKE
DINNER
DRINK
TOILET
SYMBOL
EXAMPLE BUTTON
PHOTO
Communication keys—these are sensitive touch-panel buttons. On being triggered, a recorded message related to that key is output from the speaker
In addition, symbols and photos familiar to the user can be used on the keypads to enable usability of device to be immediate in the case of some individuals
Battery indicator shows amount of battery left before recharging is required
Battery Indicator
Communication Mate
Amplified speaker provides excellent output
Ring attachment for belt/trousers. This enables the device to hang from a person's trouser/belt in a similar way to a key ring

Source: Reprinted by permission of Sigil Khwaja.

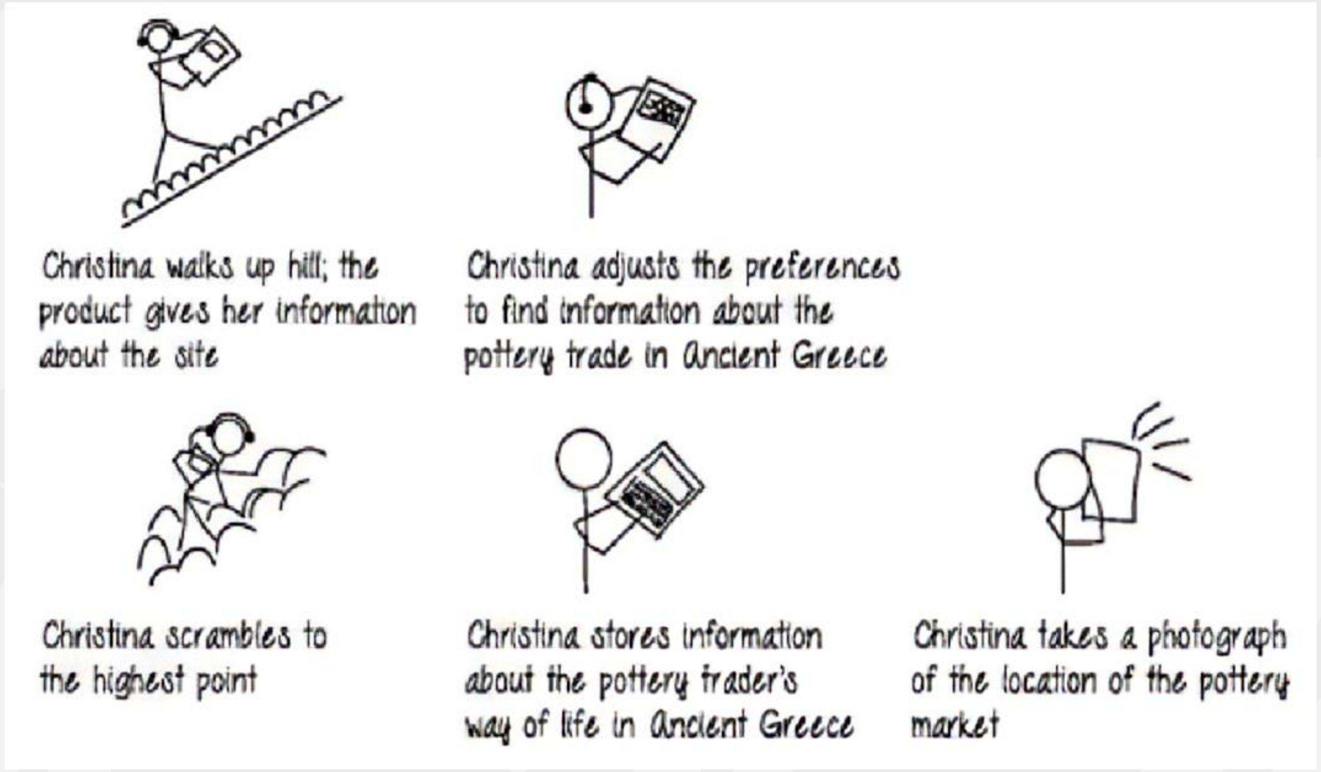
UNIVERSITY OF PLYMOUTH

Storyboarding

- Storyboarding is one example of low fidelity prototyping that is often used in conjunction with scenarios.
- A storyboard consists of a series of sketches showing how a user might progress through a task using the product under development.
- It can be a series of *screen sketches* or a series of *scenes showing* how a user can perform a task using an interactive device.

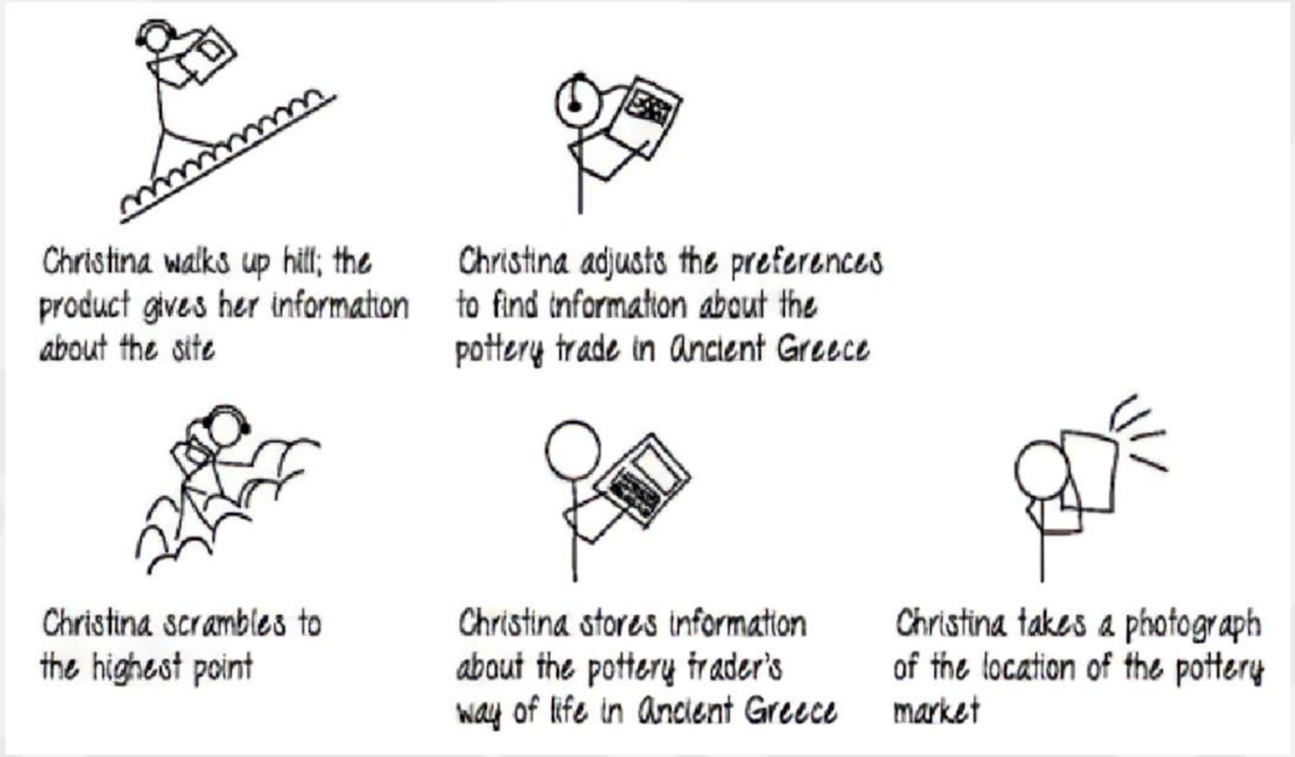
Storyboarding

- The example storyboard shown in Figure depicts a person (Christina) using a new mobile device for exploring historical sites.

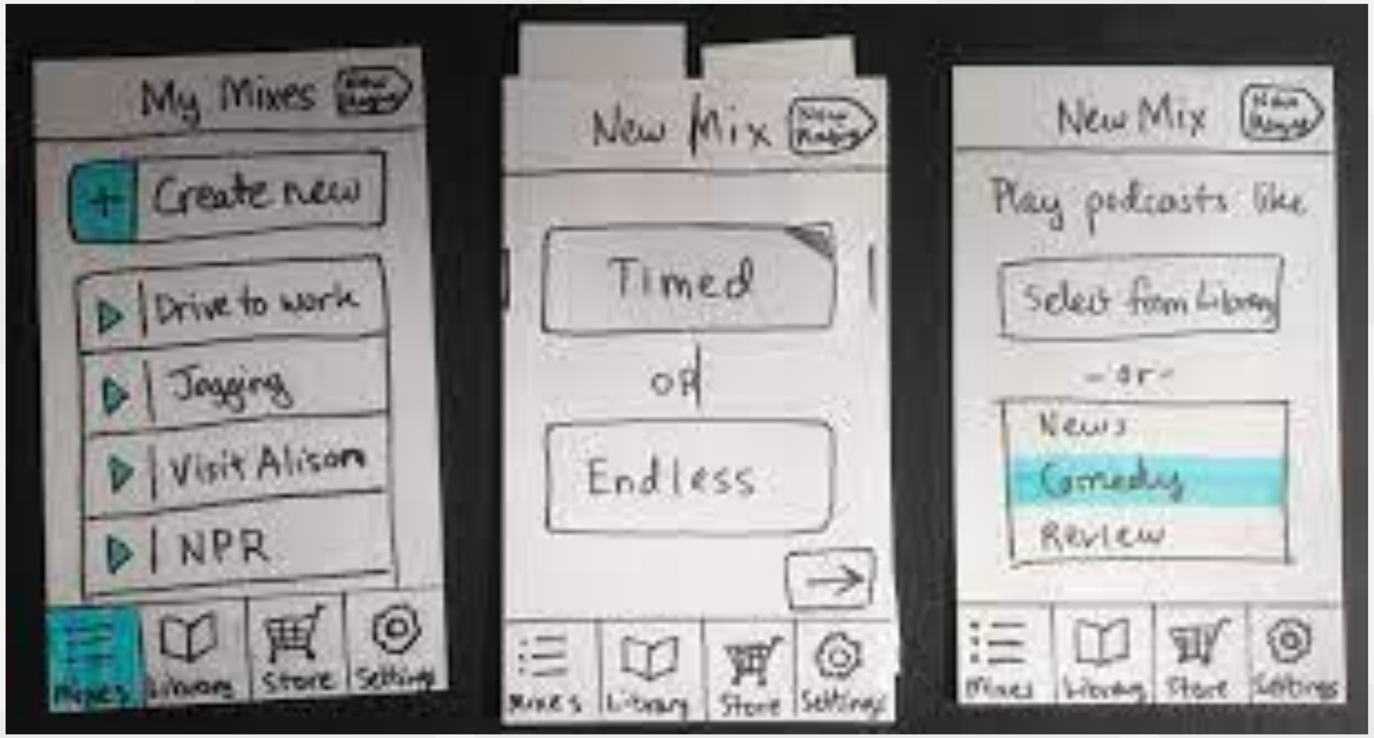

Christina walks up hill; the product gives her information about the site
Christina adjusts the preferences to find information about the pottery trade in Ancient Greece
Christina scrambles to the highest point
Christina stores information about the pottery trader's way of life in Ancient Greece
Christina takes a photograph of the location of the pottery market

Storyboarding

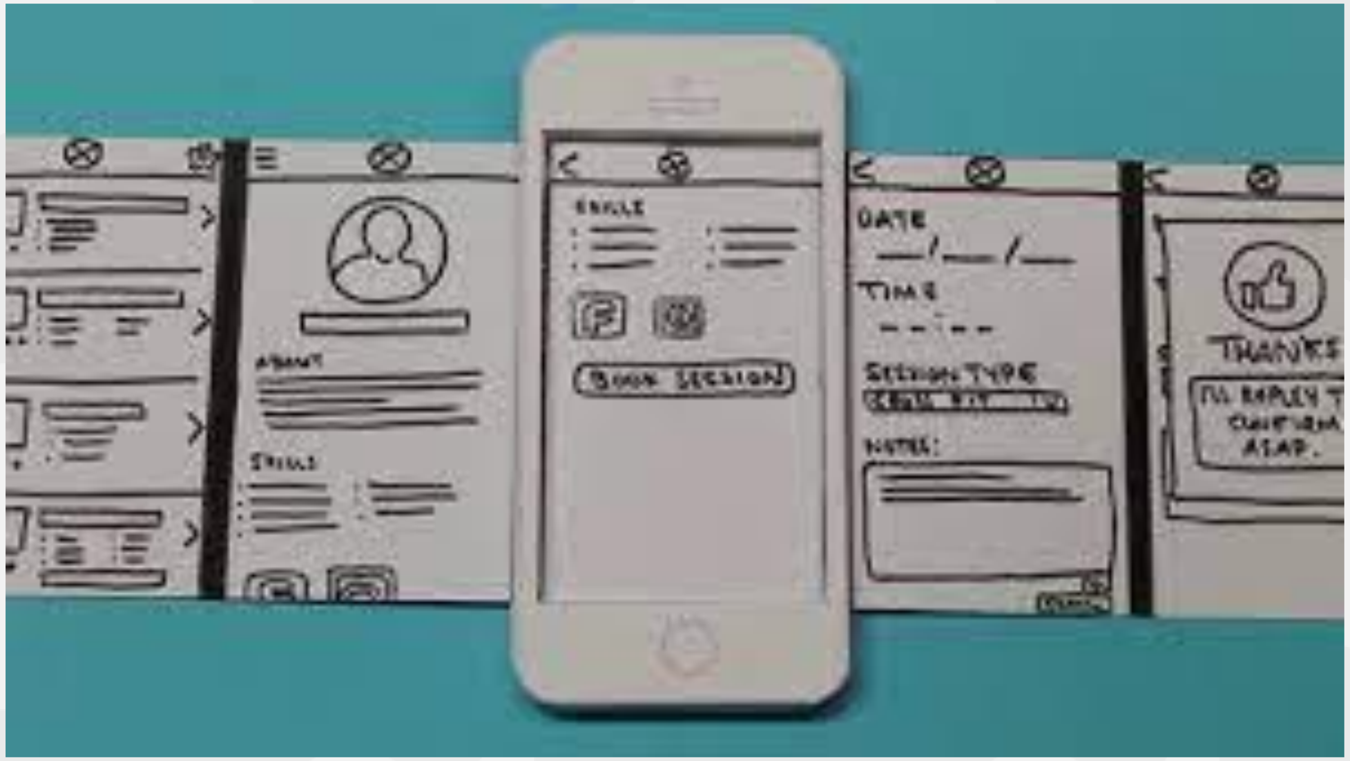
- This example shows the context of use for this device and how it might support Christina in her quest for information about the pottery trade at The Acropolis in Ancient Greece


Christina walks up hill; the product gives her information about the site
Christina adjusts the preferences to find information about the pottery trade in Ancient Greece
Christina scrambles to the highest point
Christina stores information about the pottery trader's way of life in Ancient Greece
Christina takes a photograph of the location of the pottery market

Storyboard example for Android app



Storyboard example for Android app



Sketching

- Low fidelity prototyping often relies on hand drawn sketches, and many people find it difficult to engage in this activity because they are inhibited about the quality of their drawing,
- but as Greenberg et al (2012) put it, “Sketching is not about drawing. Rather, it is about design” (p. 7)
- You can get over any inhibition by devising your own symbols and icons and practicing them referred to by Greenberg et al as a ‘sketching vocabulary’ (p.85).
- They don't have to be anything more than simple boxes, stick figures, and stars.
- Elements you might require in a storyboard sketch, for example, include digital devices, people, emotions, tables, books, etc., and actions such as give, find, transfer, and write
- If you are sketching an interface design, then you might need to draw various icons, dialog boxes, and so

Some simple examples

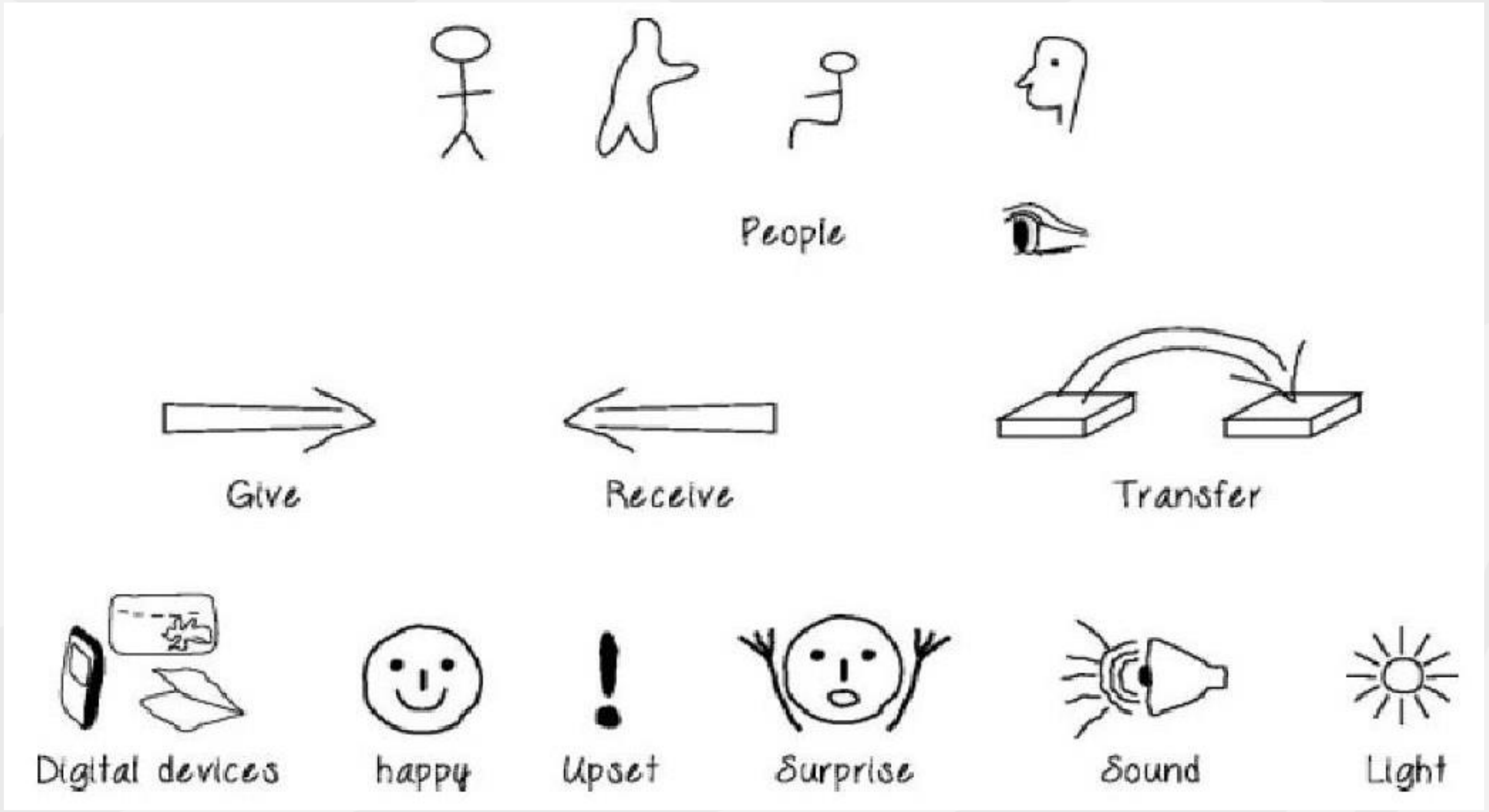


Figure 4
Some simple sketches for low fidelity prototyping

UNIVERSITY OF PLYMOUTH

- **Prototyping with index cards.**
- Using index cards (small pieces of cardboard about 3 × 5 inches) is a successful and simple way to prototype an interaction, and is used often when developing websites.
- Each card represents a screen or one element of the interaction.
- In user evaluations, the user can step through the cards, pretending to perform the task while interacting with the cards.

- **Wizard of Oz.**
- Another low-fidelity prototyping method called Wizard of Oz assumes that you have a software-based prototype.
- In this technique, the user interacts with the software as though interacting with the product. In fact, however, a human operator simulates the software's response to the user.
- The method takes its name from the classic story of the little girl who is swept away in a storm and finds herself in the Land of Oz (Baum and Denslow, 1900).
- The Wizard of Oz is a small shy man who operates a large artificial image of himself from behind a screen where no one can see him.

An example of Wizard of Oz

https://www.youtube.com/watch?v=zOZkf70kUtc

High-Fidelity Prototyping

- A high-fidelity prototype looks like the final product and/or provides more functionality than a low-fidelity prototype.
- For example, a prototype of a software system developed in Visual Basic is higher fidelity than a paper-based mockup.
- High-fidelity prototyping is useful for selling ideas to people and for testing out technical issues.
- High-fidelity prototypes can be developed by modifying and integrating existing components - both hardware and software.
- In robotics this approach has been called tinkering (Hendriks-Jansen, 1996)
- while in software development it has been referred to as Opportunistic System Development (Ncube et al, 2008).

UNIVERSITY OF PLYMOUTH

Compromises in Prototyping

- By their very nature, prototypes involve compromises: the intention is to produce something quickly to test an aspect of the product.
- Lim et al (2008) suggest an anatomy of prototyping that structures the different aspects of a prototype and what it aims to achieve.
- The kind of questions that any one prototype can answer is limited, and the prototype must be built with the key issues in mind.
- In low-fidelity prototyping, it is fairly clear that compromises have been made.
- For example, with a paper-based prototype an obvious compromise is that the device doesn't actually work!

UNIVERSITY OF PLYMOUTH

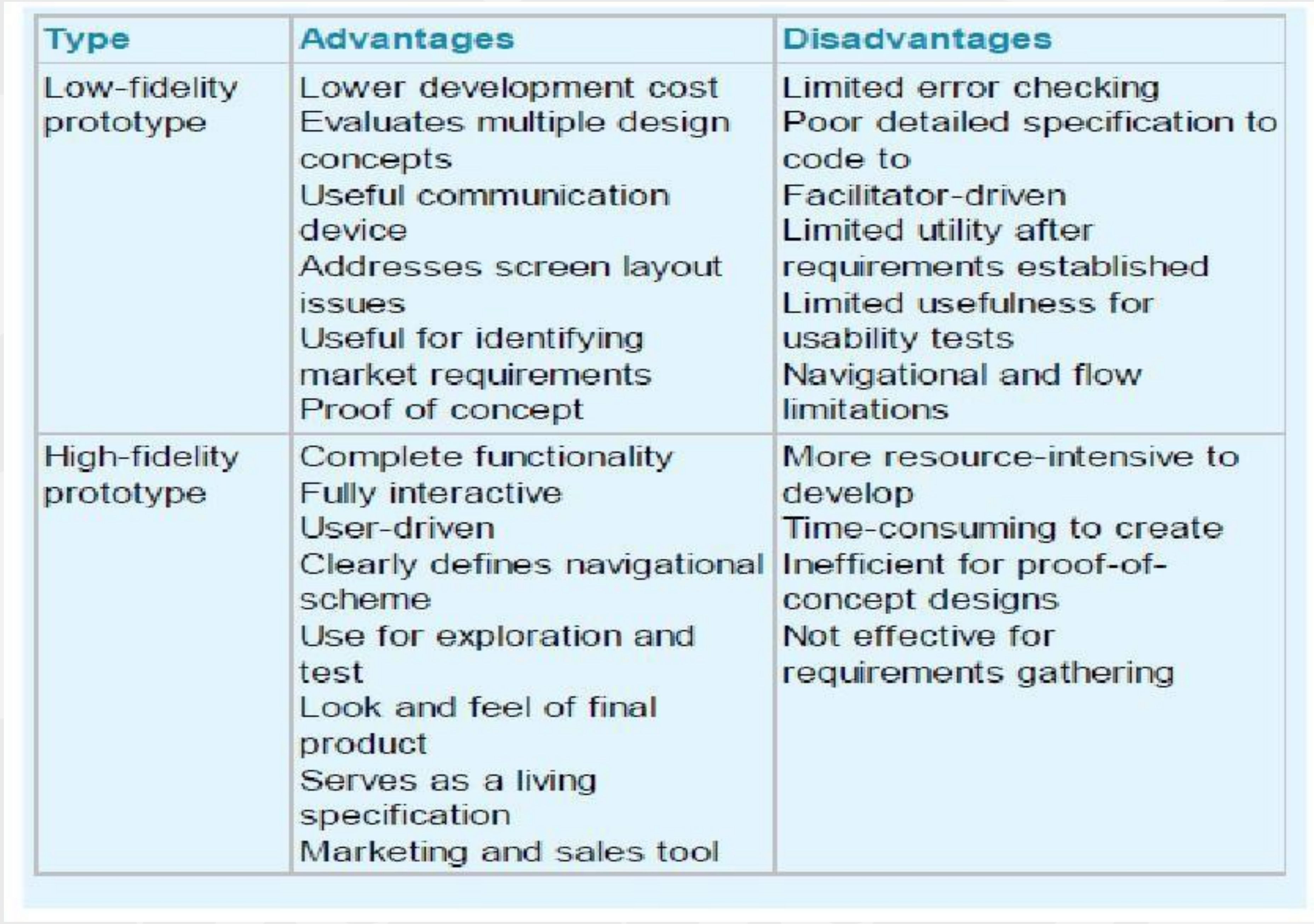
- For software-based prototyping, some of the compromises will still be fairly clear;
- for example, the response speed may be slow, or the look and feel may not be finalised, or only a limited amount of functionality may be available.
- Two common compromises that often must be traded against each other are breadth of functionality provided versus depth.

- These two kinds of prototyping are called horizontal prototyping (providing a wide range of functions but with little detail) and vertical prototyping (providing a lot of detail for only a few functions).
- Other compromises won't be obvious to a user of the system.
- For example, the internal structure of the product may not have been carefully designed, and the prototype may contain spaghetti code or be badly partitioned.
- One of the dangers of producing functional prototypes, i.e. ones that users can interact with automatically, is that the prototype can appear to be the final product.
- Another is that developers may consider fewer alternatives because the prototype works and users like it.

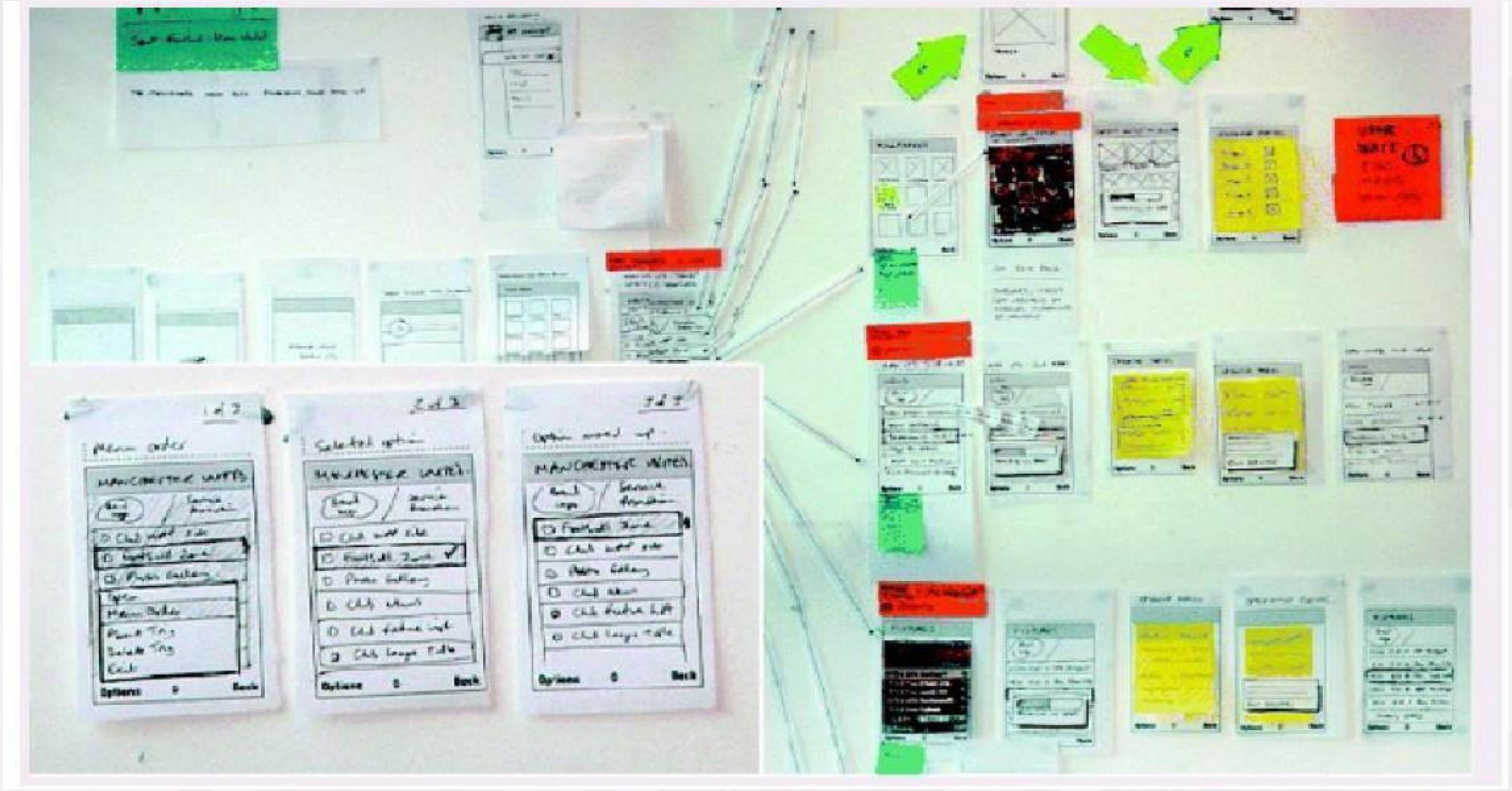
UNIVERSITY OF PLYMOUTH

"THEN IN HERE WE DO A CLAY MOCK-UP OF THE COMPUTER MODEL"

Table 1 Advantages and disadvantages of low- and highfidelity prototypes



Type	Advantages	Disadvantages
Low-fidelity prototype	Lower development cost Evaluates multiple design concepts Useful communication device Addresses screen layout issues Useful for identifying market requirements Proof of concept	Limited error checking Poor detailed specification to code to Facilitator-driven Limited utility after requirements established Limited usefulness for usability tests Navigational and flow limitations
High-fidelity prototype	Complete functionality Fully interactive User-driven Clearly defines navigational scheme Use for exploration and test Look and feel of final product Serves as a living specification Marketing and sales tool	More resource-intensive to develop Time-consuming to create Inefficient for proof-of-concept designs Not effective for requirements gathering



UNIVERSITY OF PLYMOUTH

Figure 3 Prototype developed for cell phone user interface

Unreliability of prototypes

- Although prototypes will have undergone extensive user evaluation, they will not necessarily have been subjected to rigorous quality testing for other characteristics such as robustness and error-free operation.

- Building a product to be used by thousands or millions of people running on various platforms and under a wide range of circumstances requires a different testing regime than producing a quick prototype to answer specific questions.

UNIVERSITY OF PLYMOUTH

Conceptual Design

- Conceptual design is concerned with transforming requirements into a conceptual model.
- Designing the conceptual model is fundamental to interaction design
- A conceptual model is an outline of what people can do with a product and what concepts are needed to understand how to interact with it.
- The former will emerge from the current functional requirements; possibly it will be a subset of them, possibly all of them, and possibly an extended version of them.

Conceptual Design

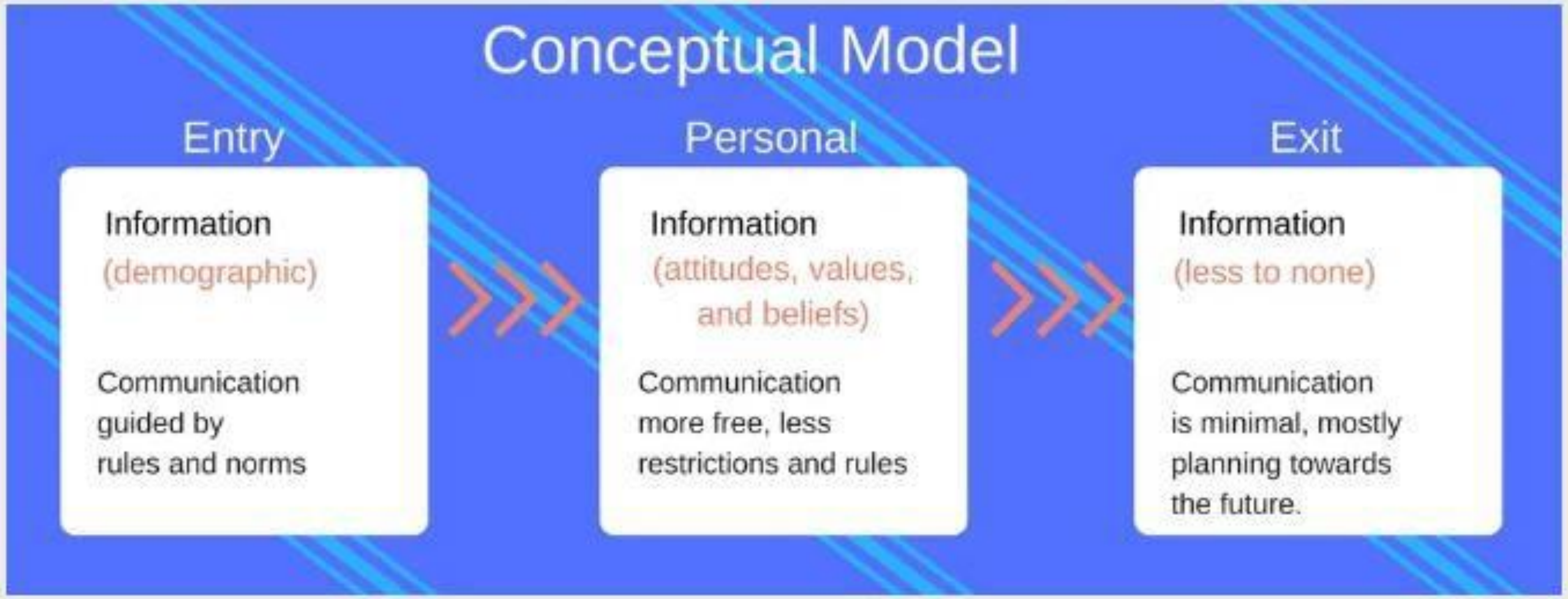
- The concepts needed to understand how to interact with the product depend on a variety of issues related to:
  - who the user will be,
  - what kind of interaction will be used,
  - what kind of interface will be used,
  - terminology, metaphors, application domain, and so on.
- The first step in getting a concrete view of the conceptual model is to analyse the data you have gathered about your users and their goals and try to walk in their shoes.

- From that, a picture of what you want the users' experience to be when using the new product will emerge and become more concrete.
- This process is helped by considering the issues in this section, and by using scenarios and prototypes to capture and experiment with ideas.
- Mood boards (traditionally used in fashion and interior design) may be used to capture the desired feel of a new product.
- This is informed by results from the requirements activity and considered in the context of technological feasibility.

UNIVERSITY OF PLYMOUTH

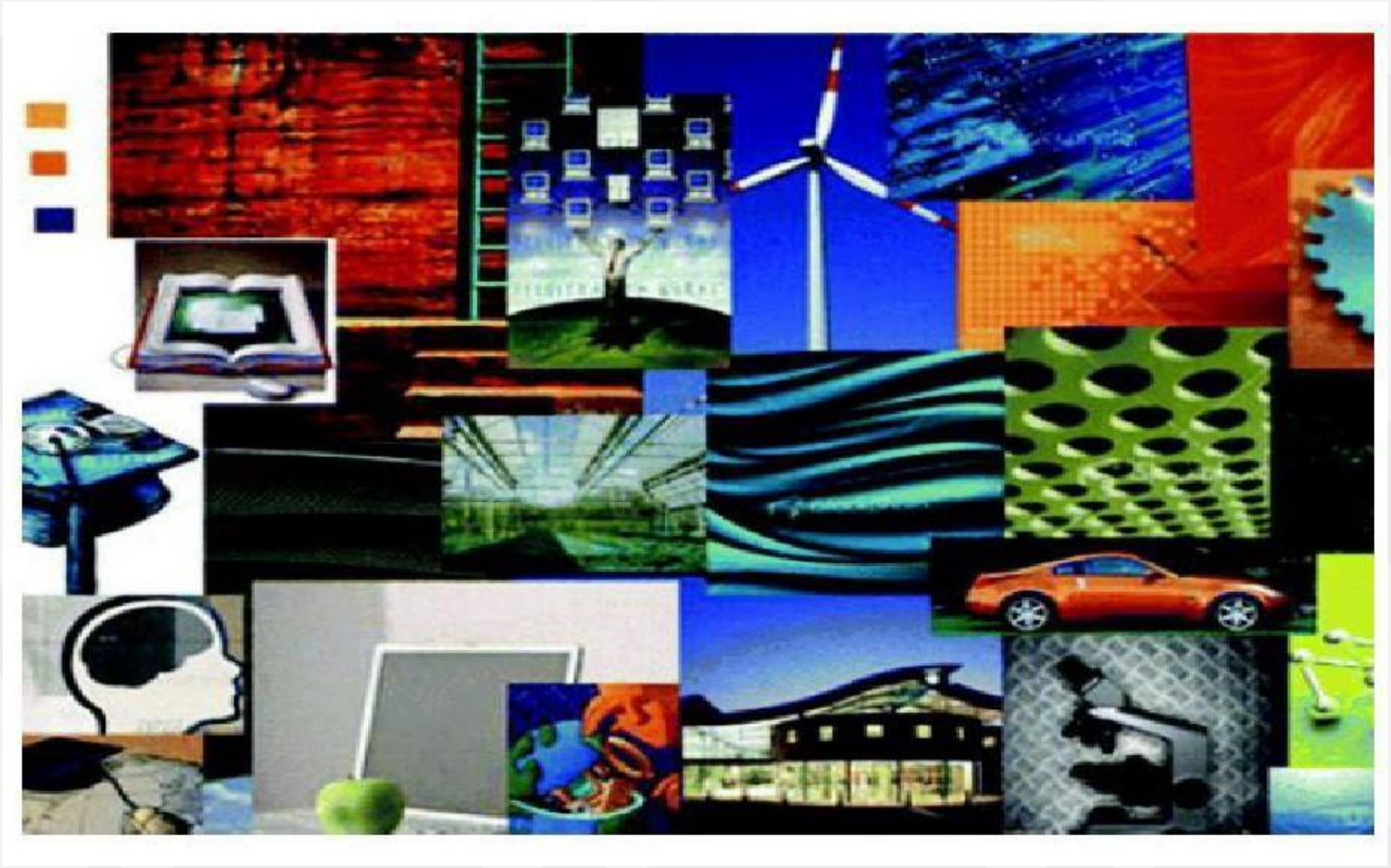
Figure: Mood board
https://www.adobe.com/express/learn/blog/ultimate-guide-mood-boards

Example: Conceptual Modelling for Mobile Applications


Conceptual Model
Entry
Information
(demographic)
Communication guided by rules and norms
Personal
Information
(attitudes, values, and beliefs)
Communication more free, less restrictions and rules
Exit
Information
(less to none)
Communication is minimal, mostly planning towards the future.

UNIVERSITY OF PLYMOUTH

Source: https://www.interaction-design.org/literature/topics/conceptual-models



UNIVERSITY OF PLYMOUTH

Figure 4 An example mood board
Source: Image courtesy of The Blog Studio www.theblogstudio.com.

- There are different ways to achieve empathy with users.
- For example, Beyer and Holtzblatt (1998), in their method Contextual Design, recommend holding review meetings within the team to get different peoples' perspectives on the data and what they observed.
- This helps to deepen understanding and to expose the whole team to different aspects.
- Ideas will emerge as this extended understanding of the requirements is established, and these can be tested against other data and scenarios, discussed with other design team members, and prototyped for testing with users.

Key guiding principles of conceptual design are:

- Keep an open mind but never forget the users and their context.
- Discuss ideas with other stakeholders as much as possible.
- Use prototyping to get rapid feedback.
- Iterate, iterate, and iterate.

Developing an Initial Conceptual Model

- Some elements of a conceptual model will derive from the requirements for the product.
- For example, the requirements activity will have provided information about the concepts involved in a task and their relationships, e.g. through task descriptions and analysis.
- Immersion in the data and attempting to empathize with the users as described above will, together with the requirements, provide information about the product's user experience goals, and give you a good understanding of what the product should be like.

UNIVERSITY OF PLYMOUTH

- Here we discuss approaches which help in pulling together an initial conceptual model. In particular, we consider:
    - Which interface metaphors would be suitable to help users understand the product?
    - Which interaction type(s) would best support the users' activities?
    - Do different interface types suggest alternative design insights or options?
- It is not the case that one way of approaching a conceptual design is right for one situation and wrong for another; all of these approaches provide different ways of thinking about the product and help in generating potential conceptual models.

UNIVERSITY OF PLYMOUTH

- **Interface metaphors.**
- **Interaction types.**
- **Interface types.**
- Interface metaphors combine familiar knowledge with new knowledge in a way that will help the user understand the product.
- Choosing suitable metaphors and combining new and familiar concepts requires a careful balance between utility and fun, and is based on a sound understanding of the users and their context.
- For example, consider an educational system to teach 6-year-olds mathematics.
- One possible metaphor is a classroom with a teacher standing at the blackboard. But if you consider the users of the system and what is likely to engage them, a metaphor that reminds the children of something they enjoy would be more suitable, such as a ball game, the circus, a playroom, and so on.

UNIVERSITY OF PLYMOUTH

Further reading

1. Sharp, Helen; Rogers, Yvonne and Preece, Jenny (2019). Interaction Design: Beyond Human-Computer Interaction.
2. Go through the **LinkedIn Learning - a basics course – Figma** on the COMP2000 DLE page to learn UI design

Summary

- Introduction to software development life cycle
- SD models
- Conceptual and concrete designs
- Prototyping
- Sketching and Storyboards
- Low fidelity and high fidelity prototyping

Thank you!

UNIVERSITY OF PLYMOUTH