

HCI103

Interactive technologies

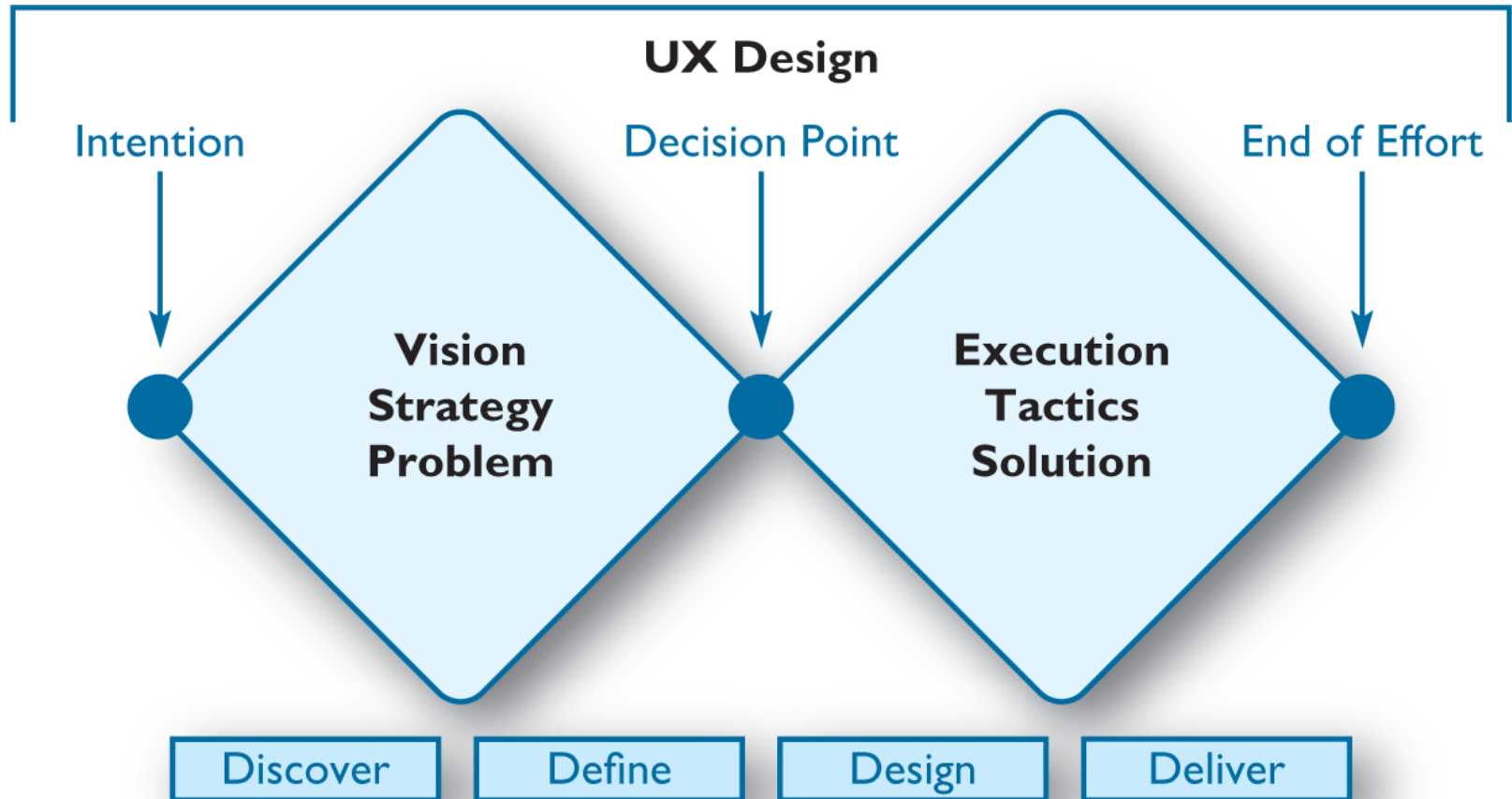
Unit 02

The design process

usability

The design process

The double diamond model



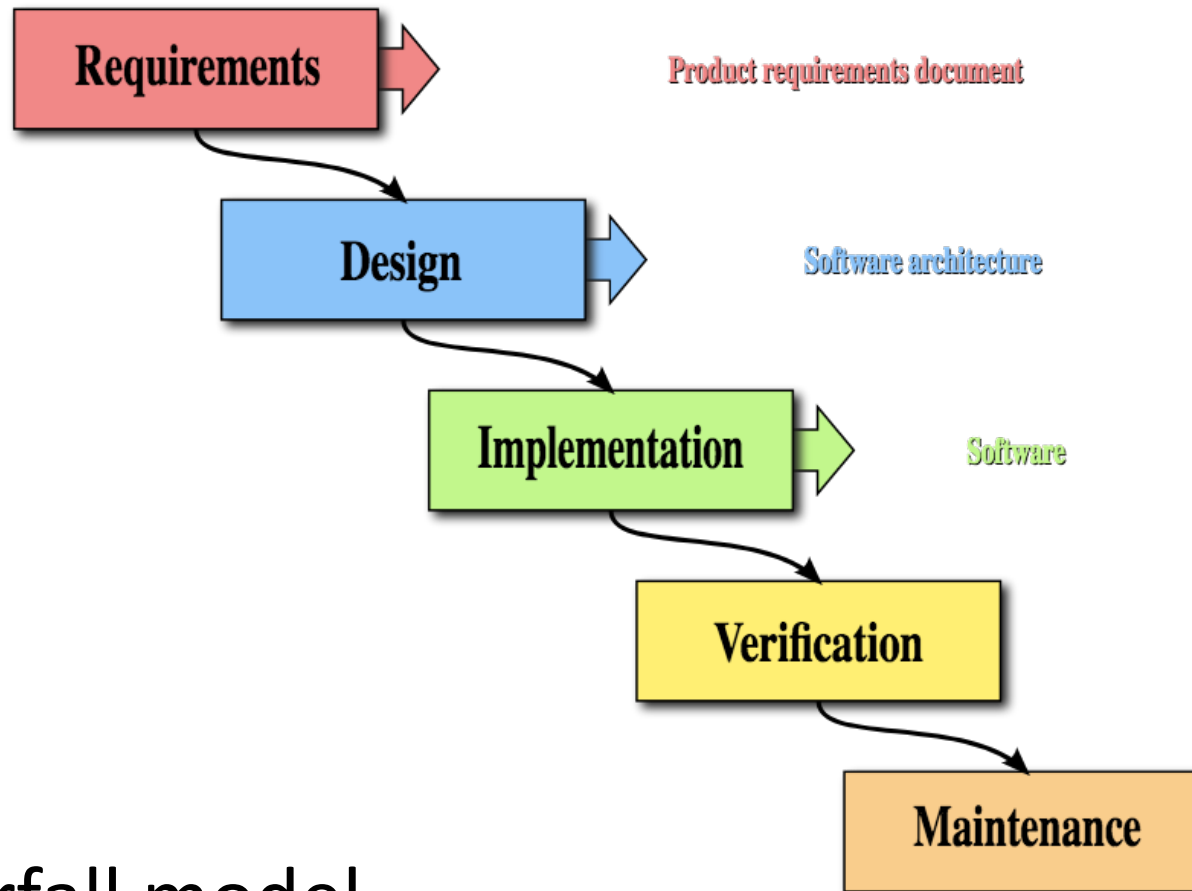
<https://www.designcouncil.org.uk/news-opinion/design-process-what-double-diamond>

Design approach

This approach has **four phases** which are iterated:

- **Discover:** Designers try to gather insights about the problem (**analysis/requirements**).
- **Define:** Designers develop a clear brief that frames the design challenge (**design**).
- **Develop:** Solutions or concepts are created, prototyped, tested, and iterated (**implement/test**).
- **Deliver:** The resulting project is finalized, produced, and launched (**deployment/**).

Design in software development



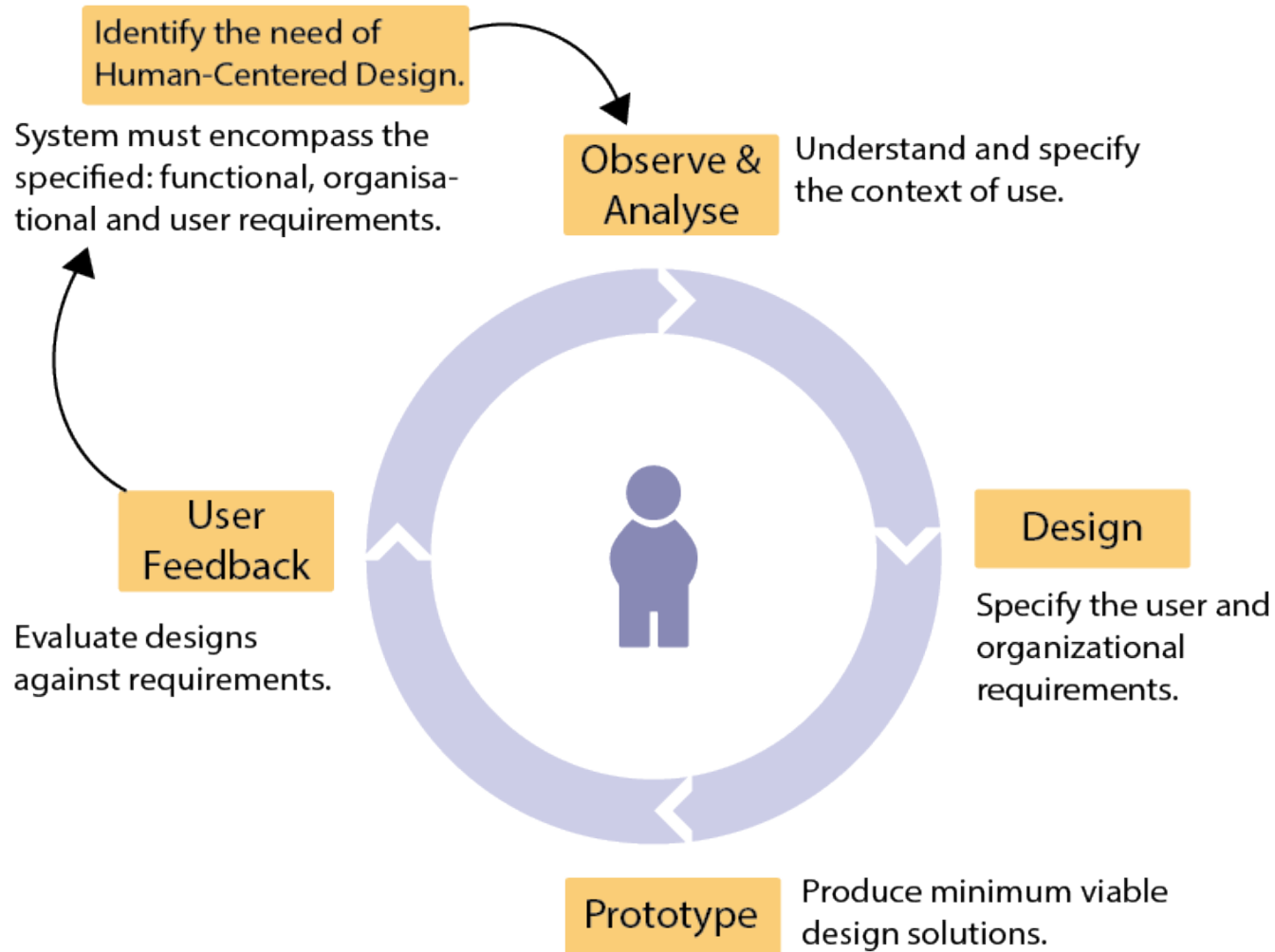
waterfall model

interaction/UX design

design activities: 1. **discover** (requirements), 2. **design** (conceptual and physical), 3. **envision** (create representations), and 4. **evaluation**.

- **Evaluation** is central to designing interactive systems. Everything gets evaluated at every step of the process.
- The process **can start at any point** – sometimes there is a conceptual design in place, sometimes we start with a prototype and sometimes we start with requirements.
- **The activities can happen in any order**, for example requirements might be evaluated, a prototype might be built and evaluated and some aspect of a physical design might then be identified.

Interaction design ISO 9241-210:2010



Understanding (analysis)

- Understanding is concerned with investigating what the system has to do, what it has to be like and how it has to meet the requirements of the product, system or service to be developed.
- Designers need to research the range of people, activities and contexts relevant to the domain they are investigating so that they can understand the requirements of the system they are developing.
- They need to understand the opportunities and constraints provided by technologies.

Requirements

- There are both **functional** and **non-functional** requirements to consider.
- **Functional requirements** are concerned with what the system should be able to do and its constraints.
- It is important for the designer to think about the whole interaction experience in an abstract way and be as **independent of current practice** as possible.
- However, there are always **functional constraints** – technical, logical and organizational, which render certain ordering, sequencing and allocation of functions inevitable.

Defining requirements

- Requirements are generated through **discussions and interactions** with people who will use or be affected by the proposed system, the **stakeholders**.
- Requirements are also generated through **observations** of existing systems, research into similar systems, what people do now and what they would like to do.
- Requirements can be generated through working with people in focus groups, design workshops and so on, where different scenarios can be considered.
- The aim is to collect and analyse the stories people have to tell. Requirements are essentially about understanding.

Stakeholders

- Stakeholders is a term that refers to **all the people who will be affected by any systems that results from the process of interactive systems design.**
- This includes the people who will be using the new system (the **'users'**) but it also includes many other people.
- For example, the organization that the system is being designed for will probably have many people in it that will not be using the system but **will be affected by it** as it might change their job.
- There may be stakeholders outside the organization such as government authorities that need to verify some procedures.
- An important part of the understanding process is to consider all the different stakeholders and how they might be affected and to decide who should be involved in discussions about the design.

Conceptual design

- Design activities concern both **conceptual** and **physical** design.
- Conceptual design includes:
 - what information and functions are needed for the system to achieve its purpose.
 - what someone will have to know to use the system.
 - finding a clear idea of the design solution and how this will be communicated to people (so that people will quickly develop a clear mental model).

Techniques for conceptual design

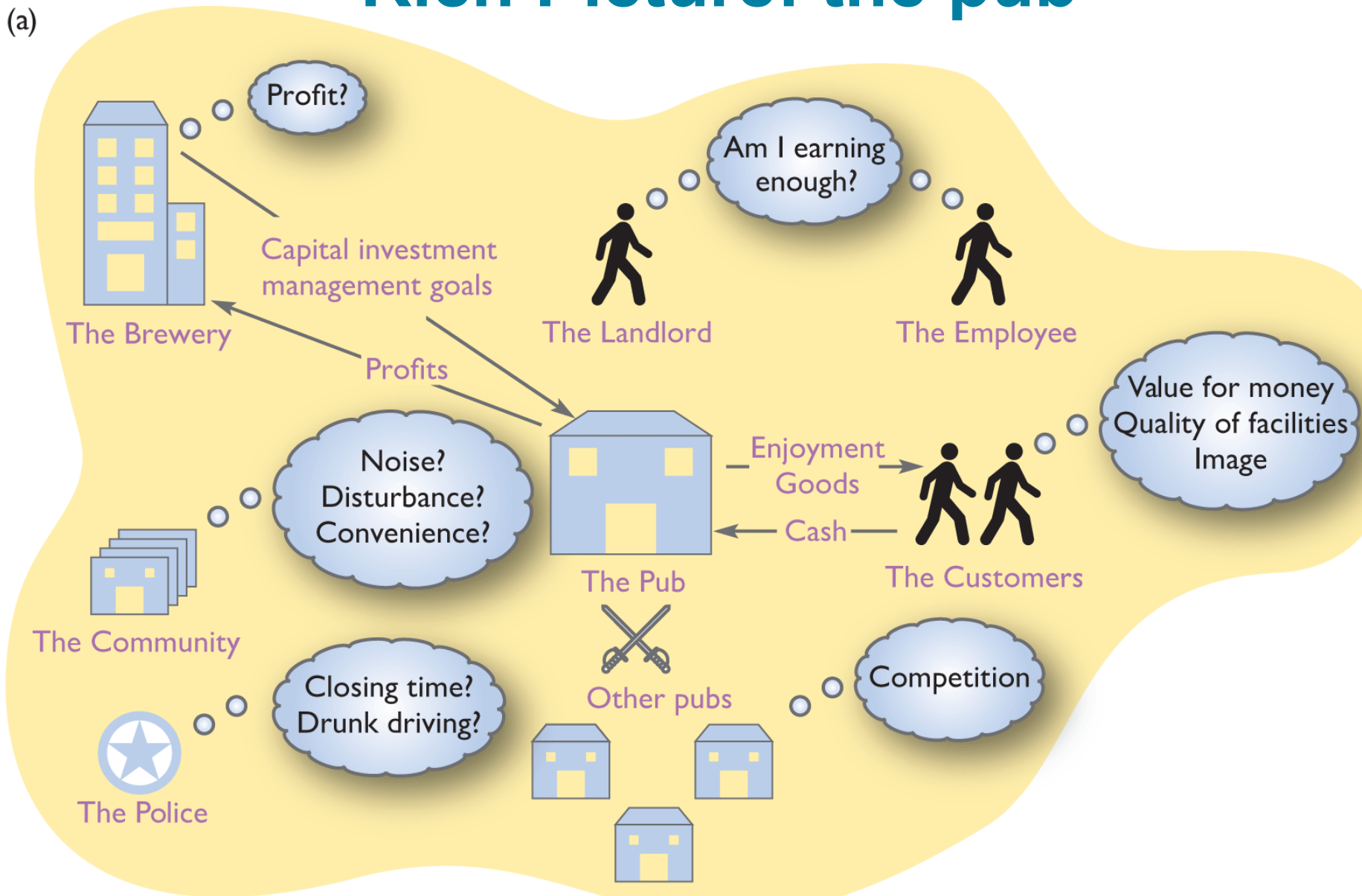
- Software engineers prefer modeling possible solutions with objects, relationships and '**use cases**' (a semi-formal scenario representation).
- Flow can be represented using **dataflow diagrams** and structure can be shown with **structure charts**.
- The conceptual design of a website, for example, will include a **site map** and a **navigation structure**.
- Many different conceptual models are used in the contextual inquiry method.

Rich Pictures

- A rich picture captures the main conceptual relationships amongst the main conceptual entities in a system – a model of the structure of a situation. Soft systems approach, (Checkland 2001), emphasizes focusing on the key transformation of a system.
- The principal stakeholders – **customers, actors** and **system owners** – should be identified.
- Most importantly, the rich picture identifies the issues or **concerns of the stakeholders**, thus helping to focus attention on problems or potential design solutions.

Rich Picture: the pub

(a)



Physical design

- Physical design is concerned with taking the abstract representation of conceptual design and translating it into concrete designs.
- Physical design is concerned with how things are going to work and with detailing the look and feel of the product, structuring interactions into logical sequences and about clarifying and presenting the **allocation of functions and knowledge between people and devices**.
- This means **requirements for hardware and software** and the **knowledge, tasks and activities** that people will be required to do.
- Physical design involves: **operational design, representational design, design of interactions**.

Operational design

- Operational design is concerned with specifying how everything works and how content is structured and stored.
- Taking a functional view of an activity means focusing on **processes** and on the **movement**, or flow, of things through a system.
- **Events** are occurrences that cause, or trigger, some other functions to be undertaken arisen from outside the system or as a result of doing something else.
- For example, some activity might be triggered on a particular day or at a particular time or by the arrival of a person or document.

Representational design

- Representational design is concerned with deciding colours, shapes, sizes and information layout.
- It is concerned with **style** and aesthetics and is particularly important for issues such as the attitudes and feelings of people but also for the efficient retrieval of information.
- Style concerns the overall ‘look and feel’ of the system.

Interaction design

- Interaction design, is concerned with the allocation of functions to human agency or to technology and with the structuring and sequencing of the interactions.
- Designers create tasks for people by the way they allocate functions.
- For example, consider the activity of making a phone call. Conceptually speaking, certain functions are necessary: indicate a desire to make a phone call, connect to the network, enter the phone number and make connection.

Envisionment

- **Designs need to be visualized** both to help designers clarify their own ideas and to enable people to evaluate them.
- Envisionment is concerned with finding appropriate media to render design ideas.
examples:
 - sketches,
 - fully functioning prototypes
 - cardboard mock-ups,
 - scenarios, (sometimes represented as storyboards).

prototyping tools: wireframe

UXPIN - GETTING STARTED

The only design platform that you will ever need.

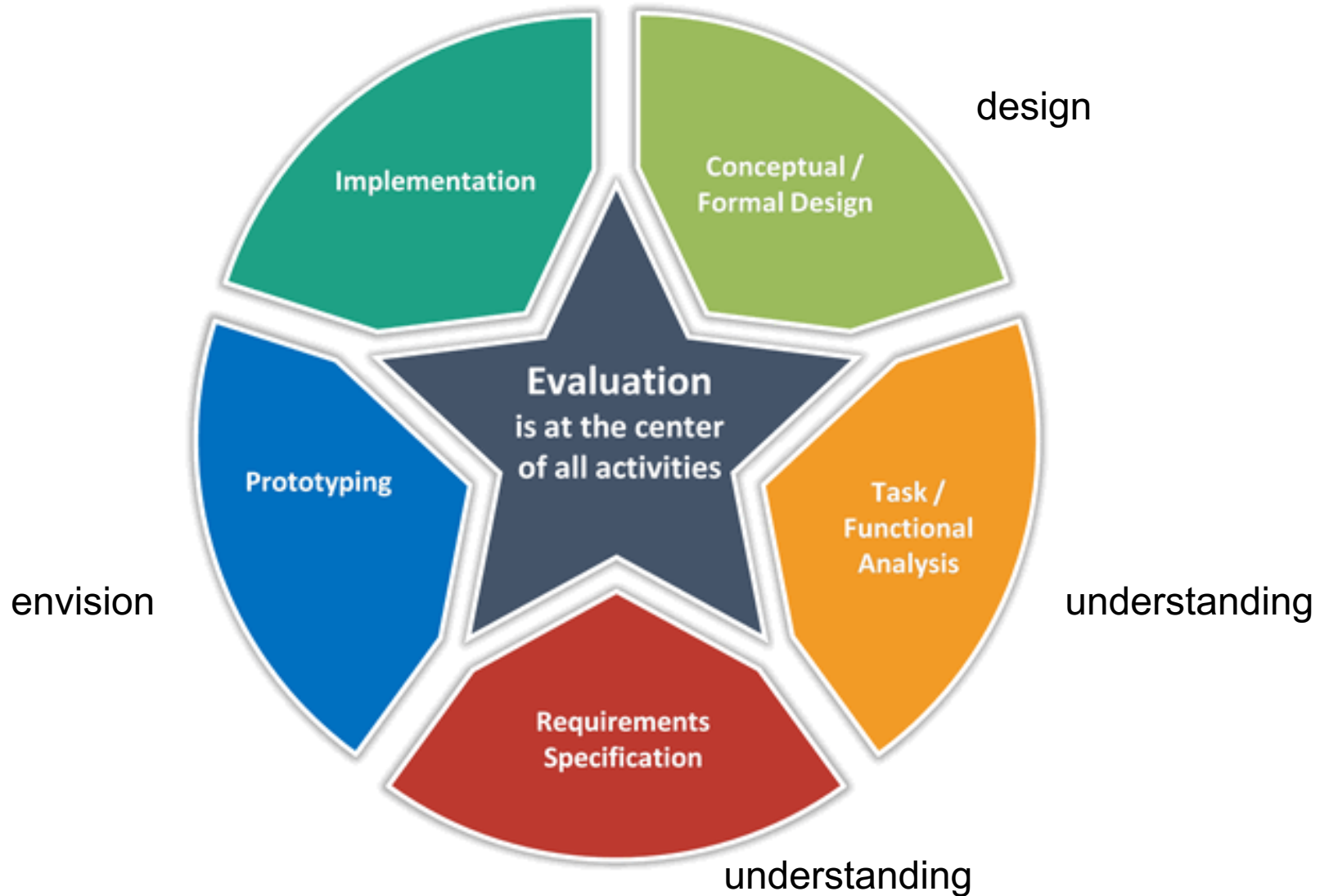
Evaluation

- Any design activity will be followed by an evaluation.
- Evaluation is tightly coupled with envisionment because the nature of the representation used will affect what can be evaluated.
- The evaluation criteria will also depend on who is able to use the representation.
- It could be a list of requirements or a high-level design brief that is sent to a client, an abstract conceptual model that is discussed with a colleague or a formal evaluation of a functional prototype by the future system users.

STAR MODEL

The Star Model (Hartson and Hix 1989)

Evaluation is in the center of design



Challenge

- Think that you plan to have a new room onto your house, or have a room converted from one use to another.
- Consider the process of this, in particular:
 - conceptual design
 - physical design
 - requirements
 - envisioned solution (e.g. prototype)

Implementation

- Ultimately, the system has to be developed and tested in order to check if it meets the requirements and finally to be ‘launched’.
- In interactive systems design, there are a variety of formal, semi-formal and informal methods of specification.

formal methods

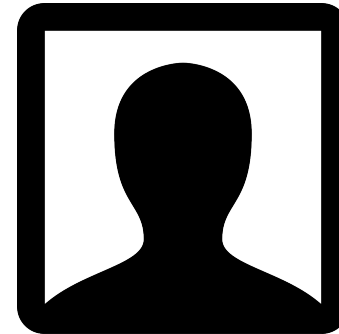
- The best known of the formal methods is the Unified Modeling Language (**UML**) (Pender, 2003).
- Over the past few years, there has been a move towards **'agile' development methods.**

Personas and scenarios in design

Personas and escenarios



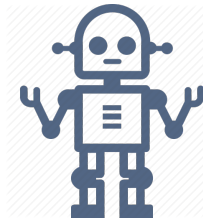
people



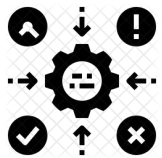
personas



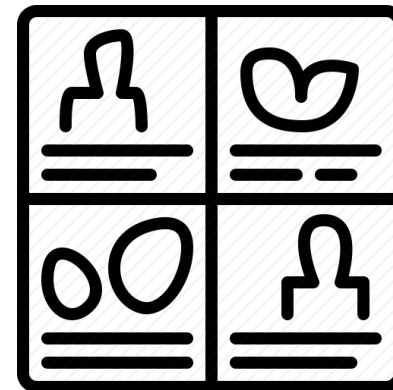
activities



technology



context



scenarios

Personas and scenarios

- In order to guide the design process, designers need to think about the People, Activities, Contexts and Technology
- The **people** who will use the system are represented by **personas**: profiles of the different types, or archetypes, of people the design is for.
- **Activities** and the **contexts** in which they will occur are envisioned through **scenarios of use**.
- Different concrete scenarios can be used **to envision how different technologies** (hardware and software) could function to achieve the overall purpose of the system and deliver a good UX.

Evolution of personas/scenarios

- **Personas** and **scenarios** are developed **early in the design process**, using the results of understanding and ideation of PACT analysis.
- Almost inevitably, **personas and scenarios evolve together** as thinking about people involves thinking about what they want to do, and thinking about activities involves thinking about who will be undertaking them!

Developing personas



- Personas are **concrete representations** of the different types of people that the system or service is being designed for.
- Personas undertake meaningful activities using the system or service that the designer will produce.
- Alan Cooper (1999, 2014) introduced the idea of personas and linked them very closely with the ideas of **goal-directed design**.
- Personas have gained rapid acceptance as a way of capturing knowledge about the people the system or service is targeted at.



Cooper (2014)

If you try to design an automobile that pleases every possible driver, you end up with a car with every possible feature that pleases nobody.



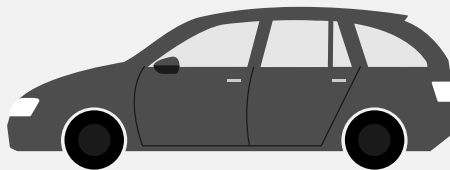
Alesandro's goals

- Go fast
- Have fun



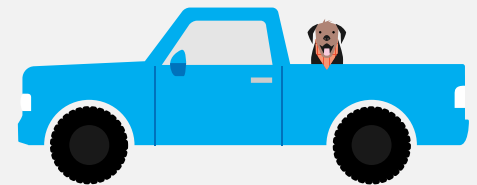
Marge's goals

- Be safe
- Be comfortable



Dale's goals

- Haul big loads
- Be reliable



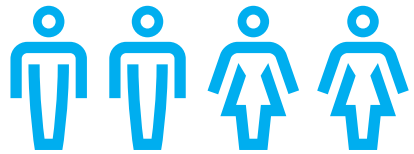
Cooper (2014)

By designing different cars for different people with different specific goals, we can create designs that other people with needs similar to our target drivers also find satisfying.

Market segments



Segment 1

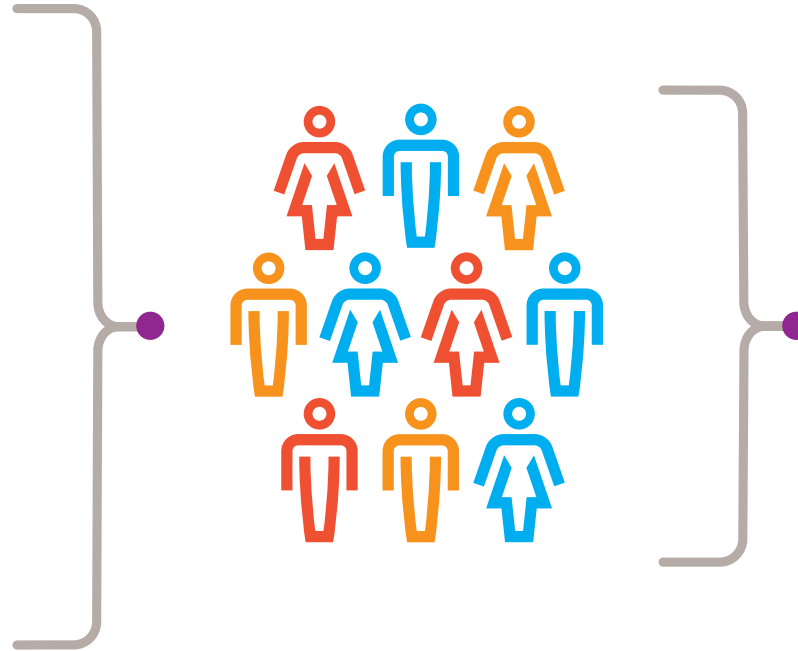


Segment 2



Segment 3

Pool of interviewees



Personas derived from behavior patterns



Kate and Sara are in Segment 1



Bob overlaps Segments 2 and 3



Ann is in Segment 3

Cooper (2014)

Market segments can be used in the Research phase to limit the range of personas to target markets. However, there is seldom a one-to-one mapping between market segments and personas.

Using personas

- In the development of personas, it is important to include the aspirations of potential users as well as looking at more functional aspects.
- Personas should help shape the whole UX and people will have an emotional response to a product or service as well as a response based on pragmatic qualities.
- Thus, designing for pleasure is important and the UX designer needs to consider the **hedonic qualities of products and services**.
- Designers create personas so that they can envisage whom they are designing for.
- They create personas so that they can put themselves in other people's shoes.

Using personas

- As any new system is likely to be used by different types of people, it is important to develop several different personas.
- Such a diverse group of people have very different goals and aspirations and differ in all the ways we discussed when discussing PACT – physically, psychologically and in terms of the usage they would make of the site

Different personas

- For example, in designing a website for people interested in the author *R.L. Stevenson* we may develop personas for:
- (i) a school teacher in Germany, (ii) a university lecturer from the United Kingdom, (iii) a child in Africa and (iv) a Stevenson enthusiast from the United States of America.

Defining personas

- There is no agreed standard for defining and documenting personas.
- Cooper *et al.* (2007) emphasize the different types of goals that people have:
- **experience goals** (simple, universal, and personal)
 - Feel smart and in control
 - Have fun
 - Feel reassured about security and sensitivity
 - Feel cool or hip or relaxed
- **end goals** (user's motivation for performing the *tasks* associated with using a specific product)
 - Be aware of problems before they become critical.
 - Stay connected with friends and family.
 - Clear my to-do list by 5:00 p.m. every day.
 - Find music that I'll love.
- **life goals** (represent the user's personal aspirations that typically go beyond the context of the product being designed)
 - Live the good life.
 - Succeed in my ambitions to...

Defining personas

- By looking at the variety of user goals, through PACT analysis, designers can identify the various behavior patterns that different personas may demonstrate.
- There is a very good in-depth treatment of personas by Pruitt and Aldrin (2006) and some excellent advice and useful templates from a number of practitioners.
- There are many templates promoted by UX design agencies freely available on the internet (template by M.Arvola).

Example persona template

Foundation for personas

Goals and driving forces
Goals

Skills and knowledge
Skills/Knowledge

Experiences
Experiences


Family and contacts
Family/contacts

Likes
Likes

Dislikes
Dislikes

Habits
Habits

Quote
Quote

Portrait

Portrait

Name:
Name:

Description:
Description

Age:
Age

Role:
Role

Sex:
Sex

Income:
Income

Hometown:
Hometown

Background
Background

Project:
Date:
Researcher:

Persona and scenario sheet. Mattias Arvola 2014.

Critiquing personas

- Designers should not fall foul of simply creating stereotyped personalities or create personas that are just like themselves or their ideal partner!
- Persona development should be agile and **subject to change** and should focus on the UX aspects of interaction, rather than other aspects such as marketing.
- The personas should include snippets of the main scenarios
- It is difficult to think about personas without thinking about what activities they might want to do and why they want to do them.

Example: Companions

- There is interest in a novel form of interaction that goes under the title of ‘Companions’, intelligent, personalized, multimodal, interface to the internet.
- Companions know their ‘owners’ and adapt the interaction to personalized interests, preferences and emotional state.
- In investigating the companions concept, one may develop a number of **personas** and **scenarios**.
- A **Health and Fitness companion (HFC)**, for example, would help provide advice and companionship for people in the domain of health and fitness.

Sandy

- age 46
- drives a lot
- drinks and eats too much
- recently divorced
- children in early 20s
- had recent health scare (suspected heart attack which was actually angina)
- kids have bought him a HFC



1. We meet Sandy in a hospital room, he's being visited by his kids.
2. They are worried about his health, he does little exercise and since his wife left him his diet has become appalling.
3. They give him a HFC (what is this?!) which will combine with his current home system. They explain that it's intended to help raise his general level of fitness, monitor his health and set and maintain a healthy balanced diet.
4. They all leave the hospital and Sandy starts the configuration.
5. Being ex-army Sandy decides that a tough-love drill instructor personality would suit him best (he's on board with the fact that he needs to get healthy), so he selects Alf, a no-nonsense archetype companion character.
6. He opens his exercise regime to be accessible by his children, on their request, as he feels this will be an added incentive for him to exercise.
7. Configuration involved biometrics such as weight, height, etc., allowing Alf to suggest appropriate training and diet.
8. Its aim is to understand whether the owner is in bad condition needing to get better, wanting to maintain current health or aim for high performance.
9. Alf reprimands bad behaviour (such as buying unhealthy food), nags when he doesn't exercise, but offers positive motivation when he does.

Mari

- age 23
- aerobics instructor
- training seriously for first marathon
- her usual training partner has moved away
- she leads a wild social life and tends to burn the candle at both ends
- she's got a targeted schedule
- companion is very proactive in pace making and motivation

1. She's set up a long-term schedule with her HFC to enable her to run her first marathon in under 4 hours.
2. This includes target goals such as what times she should be running long distances by which stage of the regime.
3. The HFC adapts to maintain the regime when Mari's social circumstance impacts her ability to train.
4. If she runs too far or too fast the companion will advise that this may have a negative impact on her training and may result in potential injury.
5. Explicit instructions in real time run ('ok, now we're gonna push hard for 2 minutes....ok, well done, let's take it easy for the next 5....etc.').
6. The HFC has access to her social schedule (through social companion?) and suggests going to a party the night before a long run may not be a great idea.
7. At the actual marathon her HFC becomes a motivating force and gives her real-time advice (eg, 'there's a hill coming up, pace yourself', it knows this from a run plug-in she bought for the HFC).



Developing scenarios

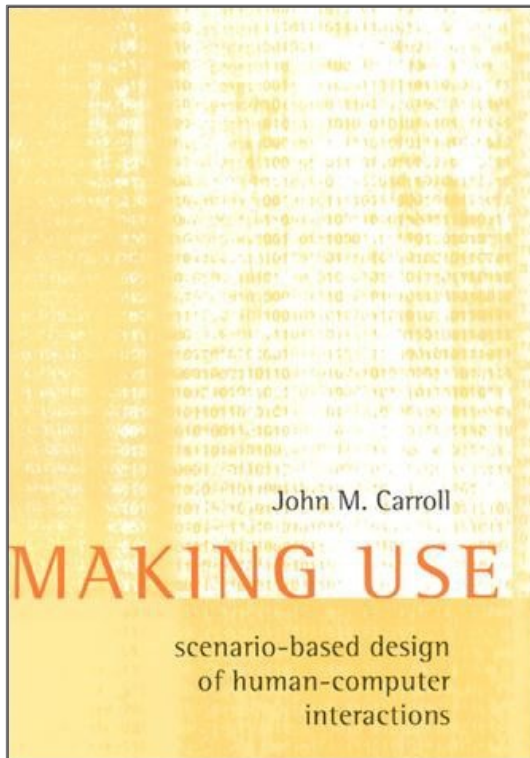
- Scenarios are stories about people undertaking activities in contexts using technologies.
- They appear in a variety of forms throughout interactive systems design and are a key component of many approaches to design.
- Scenarios have been used in software engineering, interactive systems design and human–computer interaction work for many years.
- More recently, scenario-based design has emerged as an important approach to the design of interactive systems.



Developing Scenarios

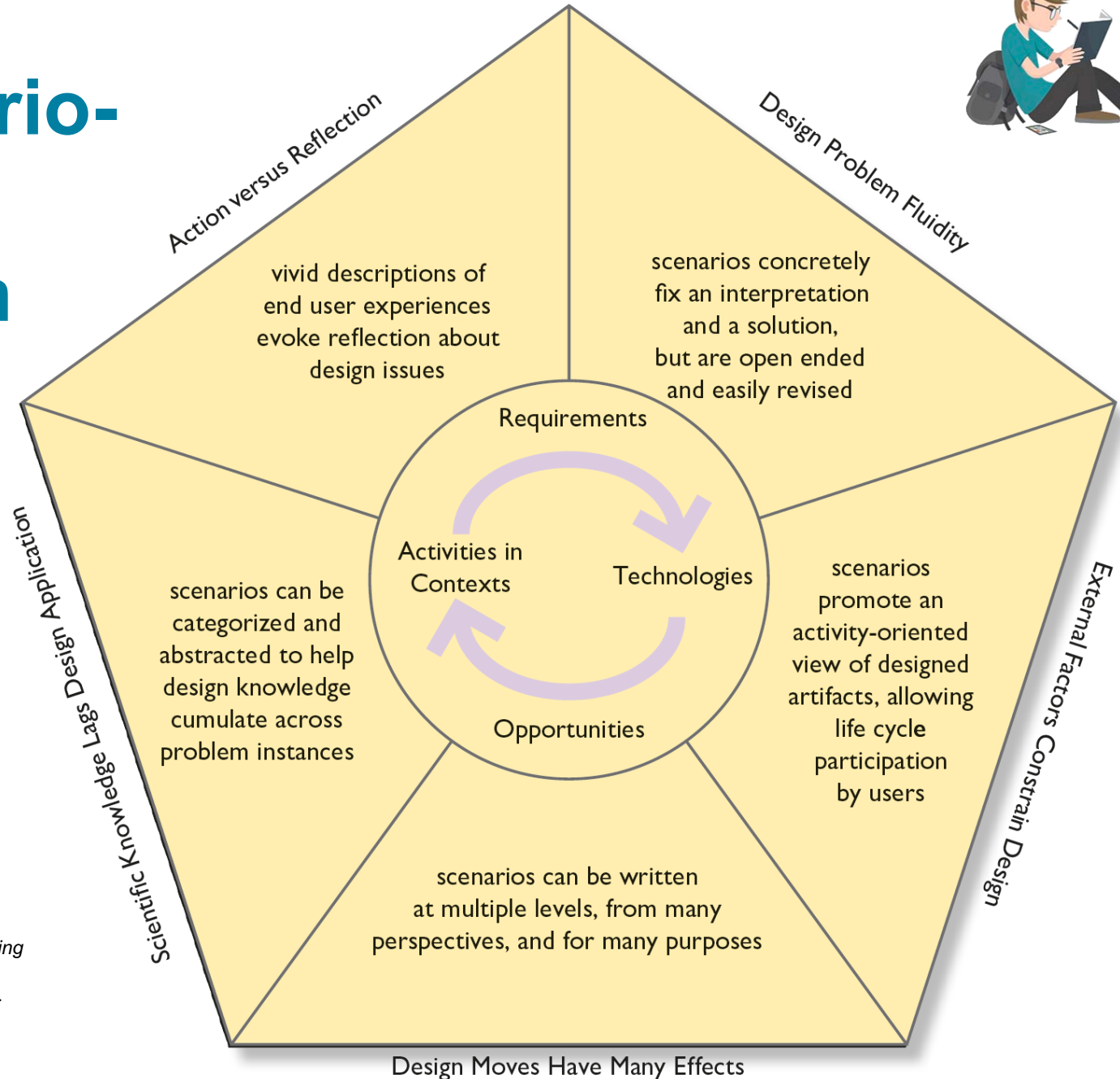
- Scenarios help designers identify and develop correct problem requirements by being at once **concrete** and **flexible**.
- Scenarios help designers to see their work as **artifacts-in-use** and, through this focus, to manage external constraints in the design process.

Developing scenarios



J. Carroll (2000) in *Making Use* illustrates how scenarios are used to deal with the inherent difficulty of doing design.

Scenario-based design



Source: After John M. Carroll. *Making Use: Scenario-based Design of Human-Computer Interactions*. Fig. 3.2, p. 69 © 2000 Massachusetts Institute of Technology, The MIT Press

Trade-offs (Rosson and Carroll, 2002)

- Rosson and Carroll (2002) describe an approach to scenario-based design in which scenarios are used throughout the design process and how they help designers to justify the claims that they make about design issues.
- Design is characterized by trade-offs. There is rarely a simple solution to a problem that solves all the issues. The adoption of one design will mean that something else cannot be achieved.
- Designers need to document their design decisions so that the trade-offs can be evaluated. Scenarios help by making the rationale for the design explicit.

Claims analysis (Rosson and Carroll, 2002)

- Claims analysis is used in identifying problems or in thinking through possible future designs (Rosson and Carroll, 2002).
- The process is simply to identify key features of a scenario and to list good and bad aspects of the design.
- Rosson and Carroll use a technique of putting a '+' beside good features and a '-' beside bad features.
- Claims analysis makes the rationale behind a design explicit.

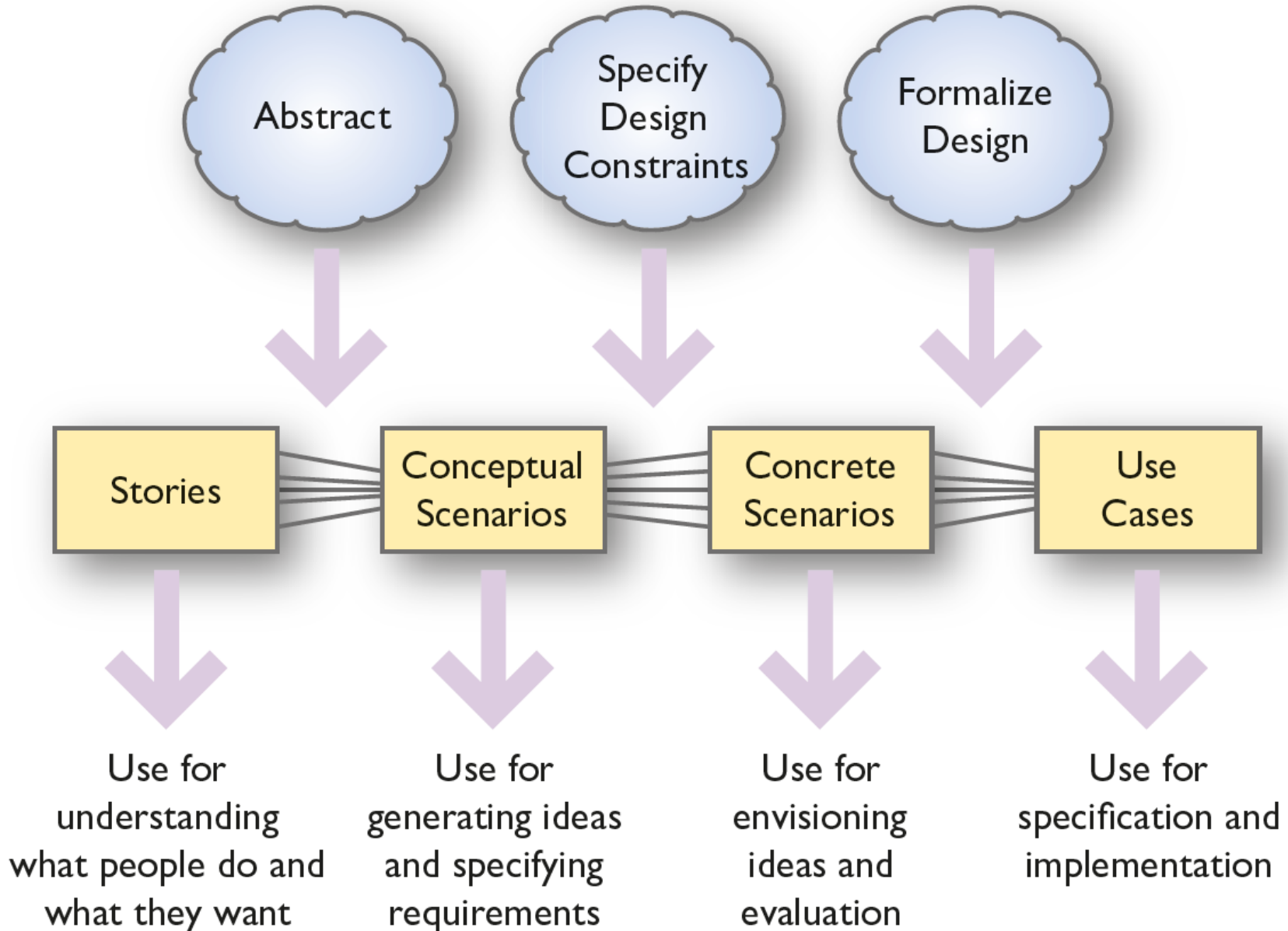
Motivations

- One central theme of the explorations concerned the motivational approaches that would be suitable for different scenarios and personas.
- The Sandy persona would need more encouragement and persuasion to exercise than the Mari persona perhaps by preventing a recorded television program from being shown until training is completed.
- Another aspect, concerning social networking, was explored through the Bjorn persona.
- Thus the personas were developed to reflect particular design issues and values.
- The whole **issue of persuasion technologies** is a difficult one for interaction design.

Using scenarios throughout design

- Scenarios (and their associated personas) are a core technique for interactive systems design. They are useful in understanding, envisioning, evaluation and conceptual/ physical design
- Different **types of scenarios**:
 - **stories**, are the real-world experiences of people
 - **conceptual scenarios**, are more abstract descriptions in which some details have been stripped away
 - **concrete scenarios**, are generated from abstract scenarios by adding specific design decisions and technologies and once completed these can be represented as use cases
 - **use cases**, are formal descriptions that can be given to programmers

Scenarios throughout design



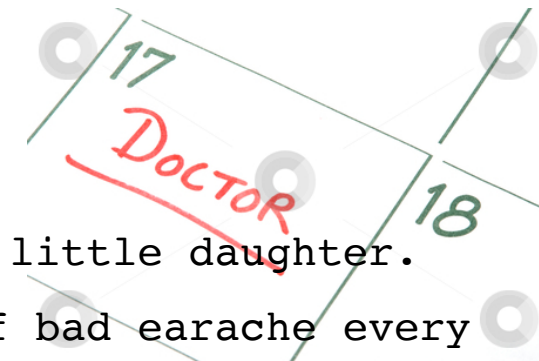
Scenarios at different stages

- At different stages of the design process, scenarios are helpful in understanding current practice and any problems or difficulties that people may be having, in generating and testing ideas, in documenting and communicating ideas to others and in evaluating designs.
- Many stories will be represented by a few conceptual scenarios. However, each conceptual scenario may generate many concrete scenarios.
- Several concrete scenarios will be represented by a single use case.
- Designers abstract from the details of stories to arrive at conceptual scenarios.
- They specify design constraints on conceptual scenarios to arrive at concrete scenarios.
- Finally, they formalize the design ideas as use cases.

Stories

- Stories are the real-world experiences, ideas, anecdotes and knowledge of people.
- These may be captured in any form and comprise small snippets of activities and the contexts in which they occur.
- This could include videos of people engaged in an activity, diary entries, photographs, documents, the results of observations and interviews and so on.
- People's stories are rich in context.
- Stories also capture many seemingly trivial details that are usually left out if people are asked to provide more formal representations of what they do.

Story: Example



I needed to make an appointment for Kirsty, my little daughter. It wasn't urgent – she had been having a lot of bad earache every time she had a cold – but I did want to see Dr Fox since she's so good with the children.

And of course ideally it had to be when Kirsty was out of school and I could take time off work.

I rang the surgery and the receptionist told me that the next appointment for Dr Fox was the next Tuesday afternoon.

That was no good since Tuesday is one of my really busy days so I asked when the next one was.

The receptionist said Thursday morning. That meant making Kirsty late for school but I agreed because they sounded very busy – the other phone kept ringing in the background – and I was in a hurry myself.

It was difficult to suggest a better time without knowing which appointments were still free.

Conceptual scenarios

- Conceptual scenarios are more abstract than stories.
- Much of the context is stripped away during the process of abstraction and **similar stories are combined together**.
- Conceptual scenarios are particularly useful for generating design ideas and for understanding the requirements of the system.
- Once the designer has accumulated a collection of stories, common elements will start to emerge.
- In this case, a number of stories such as the one of doctor's appointment, result in the conceptual scenario describing some requirements for a computerized appointments system.

Concrete scenarios

- When designers are working on a particular problem or issue, they will often identify **some feature** that applies only under certain circumstances. At this point, they may develop a **more specific elaboration of the scenario** and link it to the original. Thus, one reasonably abstract scenario may spawn several more concrete elaborations useful for exploring particular issues.
- Concrete scenarios also begin to dictate a particular interface design and a particular allocation of functions between people and devices. They are particularly useful for prototyping and envisioning design ideas and for evaluation because they are more prescriptive about some aspects of the technology. The more specific the scenario is about some aspects, the more concrete it is.

Concrete scenarios: Example

Booking an appointment/01:

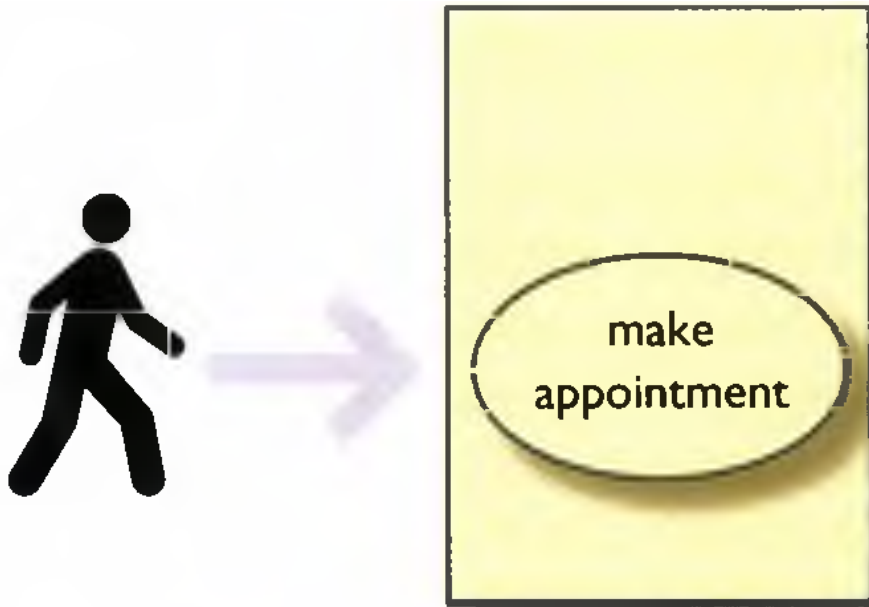
Andy Dalreach needs a doctor's appointment for his young daughter Kirsty in the next week or so. The appointment needs to be outside school time and Andy's core working hours, and ideally with Dr Fox, who is the children's specialist. Andy uses a PC and the internet at work so has no difficulty in running up the appointments booking system. He logs in and from a series of drop-down boxes chooses to have free times for Dr Fox displayed for the next two weeks

[the scenario describes next how Andy books the appointment and receives confirmation].

Use cases

- A use case describes the interaction between people (or other ‘actors’) and devices.
- It is a case of how the system is used and hence needs to describe what people do and what the system does.
- Each use case covers many slight variations in circumstances – many concrete scenarios.
- Before use cases can be specified, tasks and functions have to be allocated to humans or to the device.
- The specification of use cases both informs and is informed by the task/function allocation process.
- A set of use cases can be produced which specify the complete functionality of the system and the interactions that will occur.
- There are a number of different ways of representing use cases – from very abstract diagrams to detailed ‘pseudo code’.

Use cases



To make an appointment:

- Go to doctors' home page
- Enter username and password
- Select appointments for specific doctor Browse available dates
- Select suitable date and time
- Enter patient's name
- Click OK

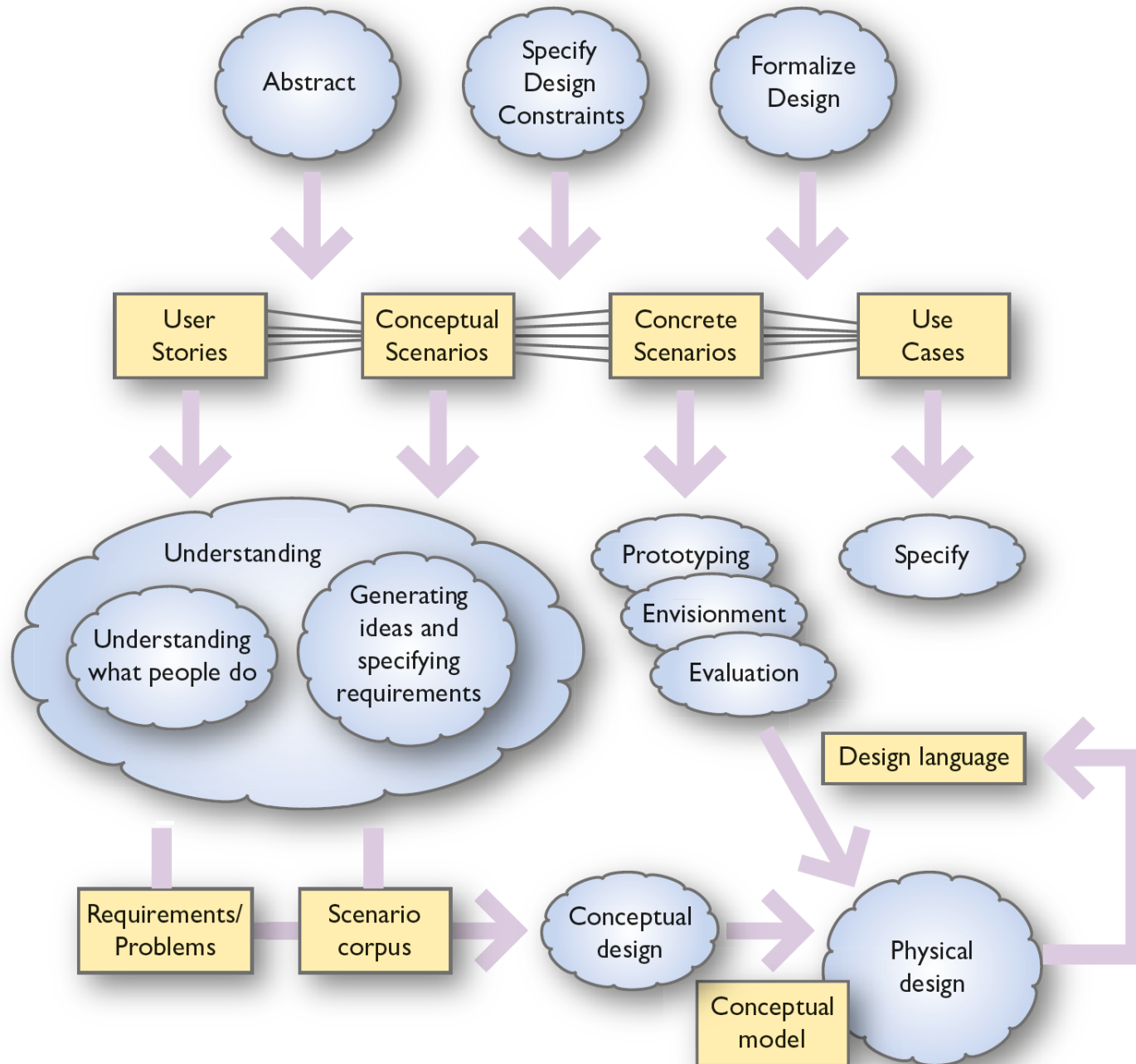
Note: Use cases are used at different levels of abstraction. Constantine and Lockwood's 'essential use cases' are similar to the conceptual scenarios described here. We reserve the term 'use case' for describing an implementable system, i.e. enough interface features have been specified, and the allocation of functions between people and the system has been completed.

A scenario-based design method

- The use of the different types of scenarios throughout design can be formalized into a scenario-based design method.
- Besides the four different types of scenario, four other artefacts are produced during the design process:
 - Requirements/Problems
 - Scenario corpus
 - Conceptual model
 - Design language.
- The specification of a system is the combination of all the different products produced during the development process.

A scenario-based design method

Clouds are processes,
Boxes are things produced



Scenarios through design

- For envisionment and evaluation, the scenarios have to be made more concrete. This means imposing design constraints.
- However, this does not mean that the designer needs to design a new concrete scenario for each possible design.
- This sort of ‘what if?’ generation and evaluation of concrete scenarios is a key aspect of design.

Requirements and problems

- In the gathering of people's stories and during the analysis and abstraction process, various issues and difficulties will come to light.
- These help the analyst/designer to establish a list of requirements – qualities or functions that any new product or system should have.
- The format of the requirements and problems is a prioritized list of issues, or a more formalized format.

Scenario corpus

- Having undertaken some analysis activities, designers will have gathered a wide range of user stories.
- Some of these will be very general and some will be quite specific, some will be fairly simple, straightforward tasks; others will be more vague.
- It is important at some point for the designer to pull these disparate experiences together in order to get a high-level, abstract view of the main activities that the product is to support.
- These conceptual scenarios will often still be grounded in a real example; we need to find an example that shares characteristics with a number of other activities.

Corpus of scenarios (notes)

- The rationale for the development of a corpus of scenarios is to uncover the 'dimensions' of the design situation and to demonstrate different aspects of those dimensions.
- Dimensions include characteristics of the various domains within which the product will operate (e.g. large and small domains, volatile or static domains and so on), the various media and data types that need to be accommodated and the characteristics of the people who will be using the system.
- The corpus of scenarios needs to cover all the main functions of the system and the events that trigger the functions.
- Different types of interaction need to be present along with any key usability issues.
- The dimensions include different types of content and how that can be structured, issues of style and aesthetics.
- The aim is to specify the scenarios at a level of abstraction that captures an appropriate level of generality that will be useful across the range of characteristics that is demonstrated within a domain.

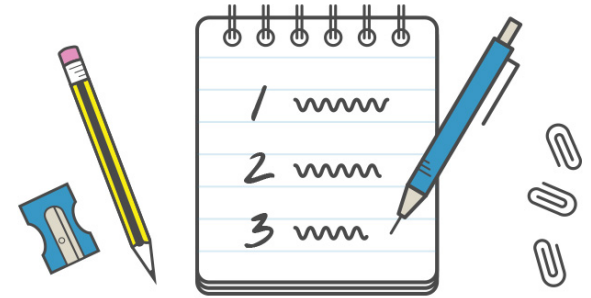
Conceptual model

- The conceptual model shows the main objects in the system, their attributes and the relationships that exist between them.
- Having a clear, well-designed conceptual model will make it easier to design so that people can develop a good, accurate mental model of the system.
- The conceptual model will also form the basis of the information architecture of a system and for any metaphor that is used in the design.
- Two famous conceptual models are the concept of the spreadsheet and the various objects such as printers, folders, documents, etc. that make up the 'desktop' metaphor of graphical interface of operating systems.

Design language

- The design language produced consists of a set of standard patterns of interaction and all the physical attributes of a design – the colours, shapes, icons and so on.
- These are brought together with the conceptual actions and objects and the ‘look and feel’ of the design is completed.
- A ‘design language’ defines the key elements of the design (such as the use of colour, style and types of buttons, sliders and other widgets and so on) and some principles and rules for putting them together.
- A consistent design language means that users need to learn only a limited number of design elements and then they can cope with a large variety of different situations.

Documenting scenarios



1. For each scenario, the designer lists the different **people** who are involved, the **activities** they are undertaking, the **contexts** of those activities and the **technologies** that are being used.
2. Each scenario should be given an **introduction**.
3. The **history and authorship** can be recorded, along with a description of how the scenario generalizes (across which domains) and the rationale for the scenario.
4. Each paragraph of each scenario **should be numbered** for ease of reference.
5. **Endnotes** are particularly useful in documenting issues raised during the development of the scenario. They are a way of capturing the claims being made about the scenarios.
6. Examples of relevant **data and media** should be collected.

Gathering data (notes)

- When working in a large design team, it is useful to accompany scenarios by real data.
- This means that different team members can share concrete examples and use these as a focus of discussion.
- Another key feature of writing scenarios is to think hard about the assumptions that are being made: to make assumptions explicit or deliberately avoid making things explicit in order to provoke debate.
- In particular, the use of personas can help focus on specific issues.
- For example, an elderly woman with arthritis might be one of the personas, thus foregrounding issues of access and the physically impaired interacting with technology.
- Finally, with these scenarios, it is important to provide a very rich context.
- The guiding principles for scenario writing are people, activities, contexts and technologies.
- The following example is grounded in a concrete example and in a specific context; this is still quite conceptual in that it is used to generate ideas and designs.

Usability

Gould and Lewis principles for human-centered design

1. Early focus on **users** and **tasks**.
2. Empirical measurement of user reactions (**evaluation**).
3. **Iterative design**.

the 4th principle:

integrated usability.

Usability

The extent to which a system, product or service can be used by **specified users** to achieve **specified goals** with **effectiveness**, **efficiency** and **satisfaction** in a **specified context of use**

The ISO 9241-210:2010 standard Ergonomics of Human-System Interaction.

Usability

- **effectiveness:** accuracy and completeness with which users achieve specified goals
- **efficiency:** resources used in relation to the results achieved (e.g. time, human effort, costs and materials)
- **satisfaction:** extent to which the user's physical, cognitive and emotional responses that result from the use of a system, product or service meet the user's needs and expectations

Usability: extending the ISO definition

- Effective to use (**effectiveness**)
- Efficient to use (**efficiency**)
- Safe to use (**safety**)
- Having good utility (**utility**)
- Easy to learn (**learnability**)
- Easy to remember how to use (**memorability**)

Related questions to ask ...

Effectiveness

- Is the product **capable of ...** allowing people to learn, carry out their work efficiently, access the information that they need, or buy the goods that they want?

Efficiency

- Once users have learned how to use a product to carry out their tasks, can they sustain a **high level of productivity?**

Safety: protecting the user from dangerous conditions

- What is the **range of errors** that are possible using the product, and what measures are there to permit users to recover easily from them?

Utility: the right kind of functionality

- Does the product provide an appropriate set of **functions** that will enable users to carry out all of their tasks in the way they want to do them?

Learnability: how easy a system is to learn to use

- Is it possible for the user to work out how to use the product by exploring the interface and trying certain actions? How hard will it be to learn the whole set of functions in this way?

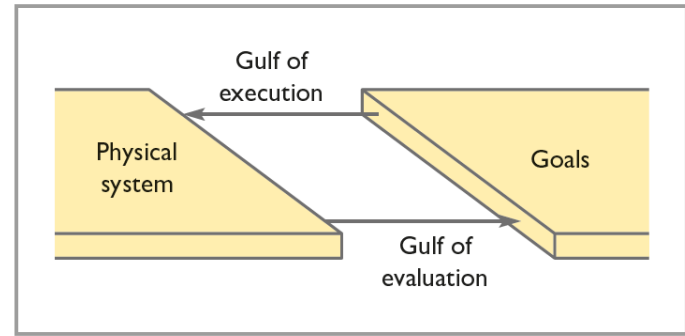
Memorability: how easy to remember to use, once learned

- What types of interface support have been provided to help users remember how to carry out tasks, especially for products and operations they use infrequently?

User experience (UX) goals

- Satisfying
- Helpful
- Fun
- Enjoyable
- Motivating
- Provocative
- Engaging
- Challenging
- Surprising
- Pleasurable
- Enhancing sociability
- Rewarding
- Exciting
- Supporting creativity
- Emotionally fulfilling
- Entertaining
- Cognitively stimulating
- Experiencing flow

The gulfs of interaction



- Don Norman (Norman, 1988) focuses on the difficulty of people having to translate their goals into the specific actions required by a user interface.
- People have goals, but devices typically only deal with simple actions. This means that two 'gulfs' have to be bridged.
- The **gulf of execution** is concerned with translating goals into actions.
- The **gulf of evaluation** is concerned with deciding whether the actions were successful in moving the person towards his or her goal.
- These gulfs have to be bridged both semantically and physically.

Affordances to bridge the gulfs of interaction



- When hammering or driving, we focus on the activity not on the technology. The technology is ‘present to hand’.
- Vermeulen, *et al.* (2013) point to the importance of good **affordances** and feedforward techniques to help bridge the gulf of execution and the importance of good feedback to bridge the gulf of evaluation.
- Winograd and Flores (1986) refer to ‘technology breakdown’ when the user becomes aware of technology. Interactive systems design aims at avoiding such breakdowns.

Accessibility

design for all

- **Accessibility** concerns removing the barriers that would otherwise exclude some people from using a service, product or system at all.
- Clearly a system must be accessible before it is usable for these users.
- A system may be assessed as highly usable according to some usability evaluation criteria but may still fail to be adopted or to satisfy some users.

Accessibility



- **Legislation** such as the United Kingdom's Equality Act 2010 and Section 508 in the United States of America now requires software to be accessible. The United Nations and the World Wide Web Consortium (W3C) have declarations and guidelines on software accessibility.
- With an increasingly wide range of computer users and technologies, designers need to focus on the demands their designs make on **people's abilities**.
- Designers have to design for the **elderly** and for **children**.
- Newell (1995) points out that the sorts of issues that face an ordinary person in an **extraordinary environment** (such as under stress, time pressures and so on) are often similar to the issues that face a person with disabilities in an ordinary environment.

kinds of exclusion

- People may be excluded from accessing interactive systems physically, conceptually, cognitively, economically, culturally, socially

inclusive design principles



- Varying ability is not a special condition of the few but a common characteristic of being human and we change physically and intellectually throughout our lives.
- If a design works well for people with disabilities, it works better for everyone.
- At any point in our lives, personal self-esteem, identity and well-being are deeply affected by our ability to function in our physical surroundings with a sense of comfort, independence and control.
- Usability and aesthetics are mutually compatible.

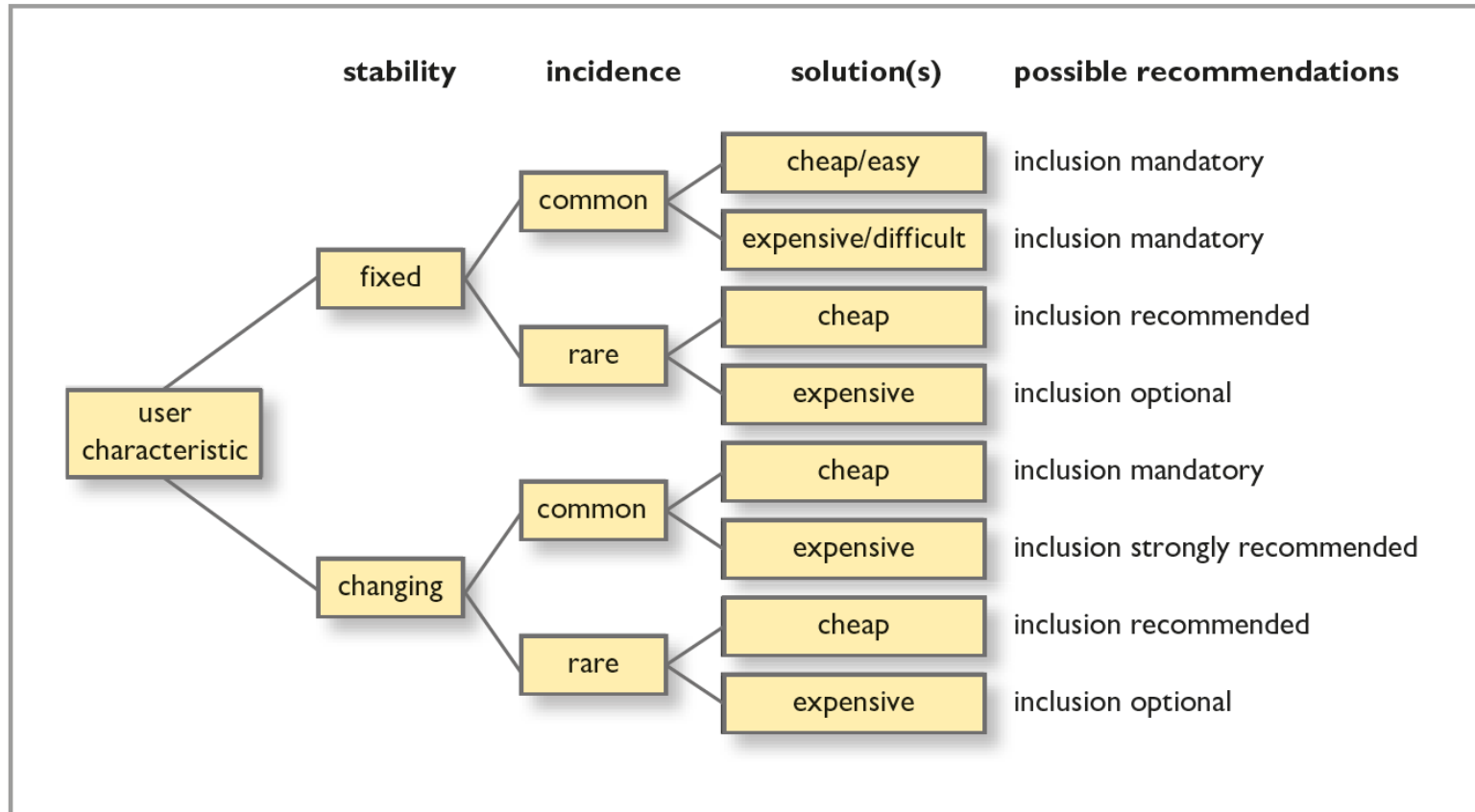
Principles of universal design

- **Equitable use.** The design does not disadvantage or stigmatize any group of users.
- **Flexibility in use.** The design accommodates a wide range of individual preferences and abilities.
- **Simple, intuitive use.** Use of the design is easy to understand, regardless of the user's experience, knowledge, language skills or current concentration level.
- **Perceptible information.** The design communicates necessary information effectively to the user, regardless of ambient conditions or the user's sensory abilities.
- **Tolerance for error.** The design minimizes hazards and the adverse consequences of accidental or unintended actions.
- **Low physical effort.** The design can be used efficiently and comfortably and with a minimum of fatigue.
- **Size and space for approach and use.** Appropriate size and space are provided for approach, reach, manipulation and use, regardless of the user's body size, posture or mobility.

Inclusive design

- Inclusive design is a more pragmatic approach that argues that there will often be reasons (e.g. technical or financial) why total inclusion is unattainable.
- Benyon *et al.* (2001) recommend undertaking an inclusivity analysis that ensures that inadvertent exclusion will be minimized and common characteristics that cause exclusion and which are relatively cheap to fix will be identified.
- Distinguishing between fixed and changing user characteristics, they present a decision tree.
- We all suffer from disabilities from time to time (e.g. a broken arm) that affect our abilities to use interactive systems, so accessibility is not something that is an issue for only a few people.

Decision tree for inclusivity analysis



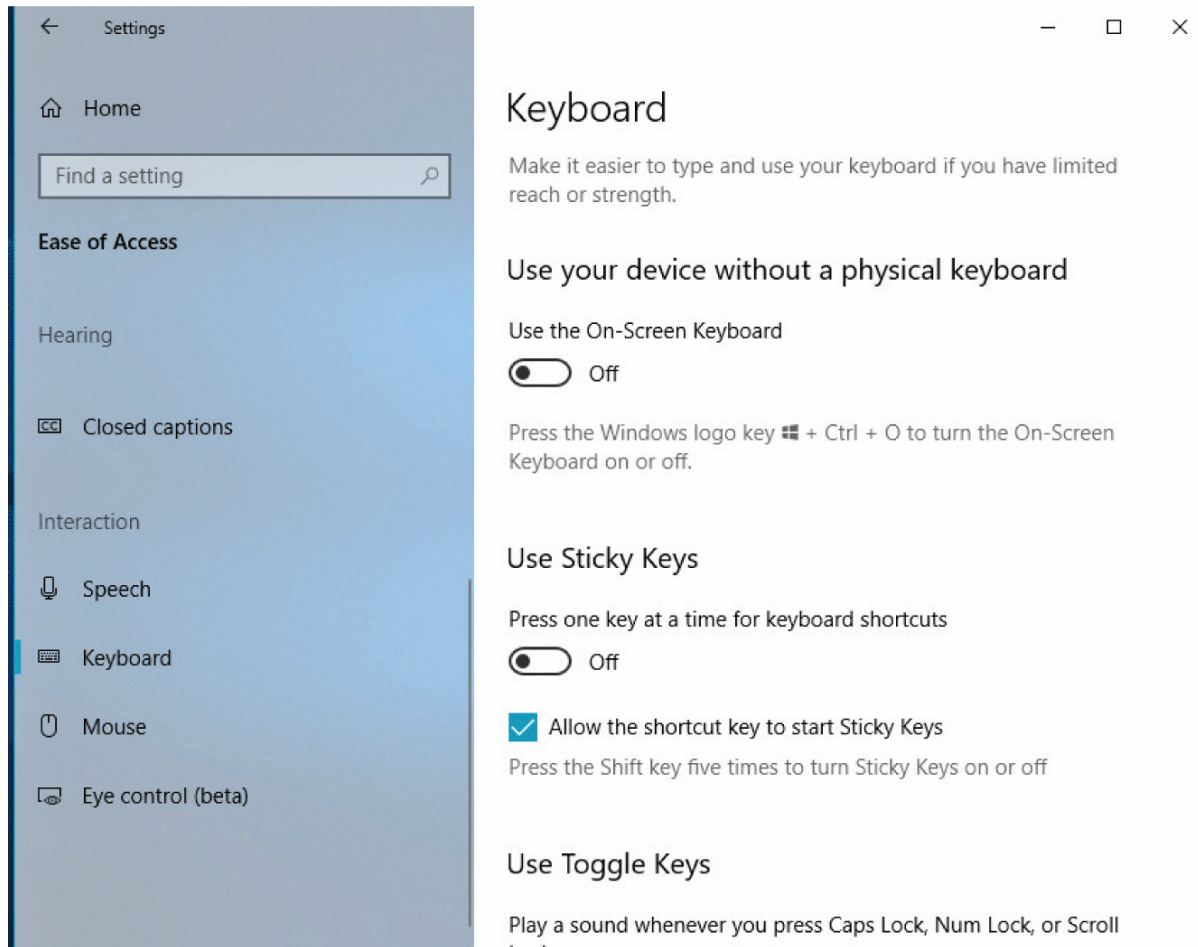
Source: after Benyon *et al.* (2001), Figure 2.3, p. 38

Inclusive design

As a way of ensuring an accessible system, designers should:

- Include people with special needs in requirements generation, user research and analysis of existing systems.
- Consider whether new features affect users with special needs (positively or negatively) and note this in the specification.
- Take account of guidelines – include evaluation against guidelines.
- Include special needs users in usability testing and beta tests.

Assistive technologies



options for accessibility keyboard settings

Web accessibility



- Web accessibility is a particularly important area as many websites exclude people who are not fit and able.
- The W4A conference and ACM's SIGACCESS group contain many specialist papers and discussions.
- The W3C web accessibility initiative (WAI) lists many automated tools that will check web pages for conformance to the W3C standards and conformance to section 508.

practical: web accessibility

Accessibility checklist

http://romeo.elsevier.com/accessibility_checklist/

Testing tools

- [WAVE](#) (available as a URL testing site, or as a Chrome/Firefox extension)
- [axe](#) (available as a Developer Tools add-on for Chrome or Firefox)
- [Accessibility Bookmarklets](#) (visually highlight headings, lists, forms, etc. on a page)
- [JAWS](#) (the most used screen reader, free trial version available)
- [NVDA](#) (free, open-source screen reader)(check Skip Donation and enter Email Address to download)

Challenge

Helping blind people learn
to code

<https://www.freecodecamp.org/news/helping-blind-people-learn-to-code-c47c68d4a237/>