Health Information Systems and Technologies:

Module background and aims

Health information systems and technologies are an integral part of today's health care delivery. As a consequence, care processes, treatment decisions and the experiences and outcomes of patients are fundamentally affected by how well these systems and technologies perform. At the same time, the technological landscape is becoming ever more complex: the number and range of systems is increasing, as is the diversity of users who rely on them, and the need for them to inter-operate and exchange data at a larger scale.

The overall aim of this module is to increase your knowledge and skills to participate in, contribute to and support the development, procurement, implementation and evaluation of health information systems and technologies. Specific aims are to:

1. Introduce you to a range of approaches and methodologies for system development and procurement, and how and when to apply them;
2. Improve your skills to (1) identify, capture and communicate requirements that fit with patient pathways and reflect the needs and expectations of a wide range of stakeholders, and (2) assess whether a system meets these requirements;
3. Develop your awareness and understanding of the technologies and architectures that underlie health information systems, and of approaches to handling their inherent complexity;
4. Explore factors and tools that are relevant for successfully planning the implementation of a system;
5. Review ways of defining and monitoring system performance.

Week 1: System Development and Procurement

Overview

- In any healthcare sector organisation (including the NHS) there are numerous information systems that are designed to manage healthcare data. These include systems that collect, store, manage and transmit a patient's electronic medical record, a hospital's operational management or a system supporting healthcare policy decisions. Health information systems also include those systems that handle data related to the activities of providers and health organisations. As an integrated effort, these may be leveraged to improve patient outcomes, inform research, and influence policy-making and decision-making. In order to deliver such systems there is a huge range of technologies.

- The health technology sector is evolving rapidly, often driven by governmental policies and by which technologies become available. This has been not just seen in the UK, but also in other countries, such as the USA, where President Bush first produced incentives for hospitals to adopt Electronic Health Records (EHRs). This development was continued by subsequent administrations. But, it must be said that health information systems are not all about technology – paper-based approaches still are in existence and do still work.

Learning outcomes

After completing week 1, you should have a better understanding of different high-level approaches to health information system development and procurement.

How this topic relates to your assignment
For your written assignment, you need to decide whether you will propose to develop/improve a health information system **in house**, or whether you will be procuring it from an **external provider**. This topic will help you to make that decision and to think through the rationale for your choice.

Materials and activities

Required material and activities are numbered to indicate the order in which you should ideally progress through them. For each activity, we have provided a recommended (but approximate) amount of time for you to spend on it.

1.1 Types of health information systems [120 mins]
When considering health information systems, it is important to look at the healthcare system as a whole, rather than focusing on just the department or organisation you work in. The whole system consists of many levels, ranging from primary care (i.e., GPs), secondary care (i.e., hospitals) and tertiary care (i.e., specialist trusts) to community care and public health. Although people have been talking about enterprise systems to support this whole system view, it is much more common that each system level –and often the organisations within it—will have their own information systems.

1.2 Approaches to system development [120 mins]
This section provides a general introduction to system development.

1.3 Procurement in the NHS [120 mins]
This section provides an introduction to procuring health information systems in the NHS

1.4 Free or Open Source Software [60 mins]
Traditionally, open source systems have been driven by academic communities. However, there are more and more healthcare organisations that are taking this approach on, including the NHS. This section gives you a brief introduction into open source programmes relevant to health information systems.

1.5 Thinking about your module assignment [120 mins]

It may seem like early days, but the sooner you start thinking about your assessment the better. This task will help you get started.

1.1 Types of health information systems

When considering health information systems, it is important to look at the **healthcare system as a whole**, rather than focusing on just the department or organisation you work in. The whole system consists of many levels, ranging from **primary care** (i.e., GPs), **secondary care** (i.e., hospitals) and **tertiary care** (i.e., specialist trusts) to **community care** and **public health**. Although people have been talking about enterprise systems to support this whole system view, it is much more common that each system level –and often the organisations within it—will have their own information systems.

A strategic perspective

Gaining a strategic perspective on health information systems means gaining insight into the needs of an organisation and its wider context in terms of managing processes, activities and information. This begins with building an **understanding of how people, systems and the environment (could potentially) interact** to deliver what is required. To start exploring this, some of the key **strategic questions** include:

- How is information managed in **complex environments** where there are many inter-related systems that serve different purposes, from clinical care to hospital management to logistics?
- What are the **technical** and **non-technical requirements** that need to be met in order for health information systems to work?
- Should we build, buy or outsource the system?
    - **Build**: the organisation develops a **bespoke product in-house**. It means maximum responsibility for, but also maximum in-house control over the development process. In Topic 1, we will introduce you to the **system development lifecycle** (**SDLC**) as an approach to system development.
    - **Buy**: the organisation buys more-or-less "shrink-wrapped" components from an **external supplier**. This means giving up some flexibility to customise the system to local needs, but at the same time allows learning from how other organisations have used the system. If an organisation chooses to buy (or outsource) a solution, you will need to consider **procurement processes**. Topic 1 will provide an overview of these processes in the NHS.
    - **Outsource**: the organisation places **contracts for service provision by a third party**. A **service level agreement** defines what level of service the supplier must provide, and what the penalties are if they fail to do so.

- What level of **technical and user support** is needed once the system has been deployed?

These are questions we will consider throughout the module.

Types of health information systems

- There are many types of information systems in use to manage healthcare data. Can you think of some examples? What information do they capture and for what purpose?
- To further introduce you to different types of health information systems, please read chapter 10 (Electronic Health Records) from the book by Enrico Coiera: A Guide to Health Informatics (Third Edition). It is available through the reading list.

Back to summary

1.2 Approaches to system development

- First, watch the video lecture (see under 'Further resources'), where Prof George Moulton explains the different approaches to system development and their benefits and limitations.
- After listening to Prof Moulton's video lecture, read the paper titled 'Agile innovation management in government: a research agenda' (see reading list).
- Do you think **Agile** is possible in the NHS and similar health systems? Share your thoughts on the forum.

Back to summary

Approaches to system development

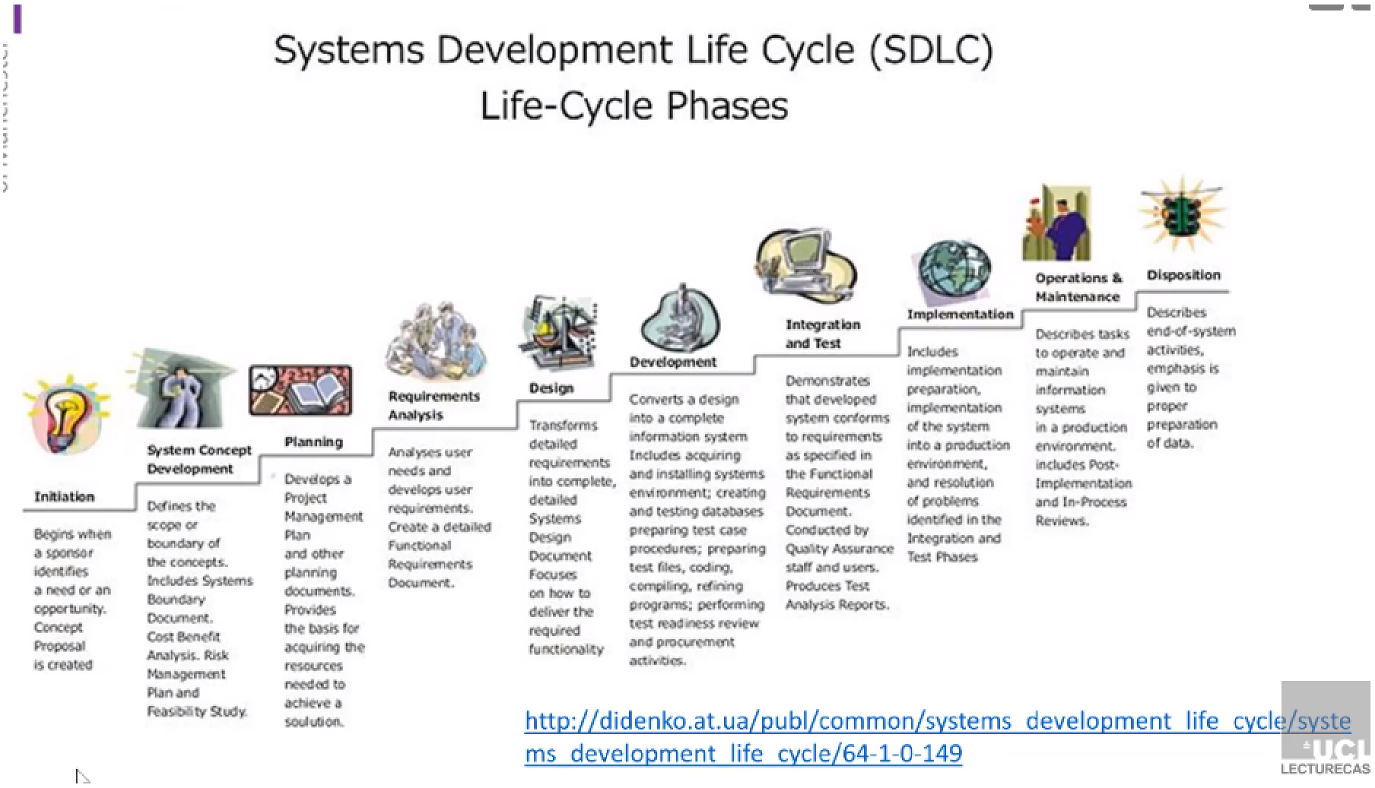
System development

- It is a process of taking a **set of business requirements** and, thorugh a **series of structured stages**, **translating them into an IT system**.
- The types of stages depend on the **development approach**
- It doesn't take place in isolation and is linked to many disciplines (system architecture, programming, testing, configuration management, quality control and quality assurance, service management, project management)

System Development Life Cycle (SDLC)

SDLC allows to think about the development of a system as a progressive process, with the system almost like a living entity.

Life-Cycle Phases



Systems Development Life Cycle (SDLC)
Life-Cycle Phases

Initiation
Begins when a sponsor identifies a need or an opportunity. Concept Proposal is created

System Concept Development
Defines the scope or boundary of the concepts. Includes Systems Boundary Document. Cost Benefit Analysis. Risk Management Plan and Feasibility Study.

Planning
Develops a Project Management Plan and other planning documents. Provides the basis for acquiring the resources needed to achieve a solution.

Requirements Analysis
Analyses user needs and develops user requirements. Create a detailed Functional Requirements Document.

Design
Transforms detailed requirements into complete, detailed Systems Design Document Focuses on how to deliver the required functionality

Development
Converts a design into a complete information system Includes acquiring and installing systems environment; creating and testing databases preparing test case procedures; preparing test files, coding, compiling, refining programs; performing test readiness review and procurement activities.

Integration and Test
Demonstrates that developed system conforms to requirements as specified in the Functional Requirements Document. Conducted by Quality Assurance staff and users. Produces Test Analysis Reports.

Implementation
Includes implementation preparation, implementation of the system into a production environment, and resolution of problems identified in the Integration and Test Phases

Operations & Maintenance
Describes tasks to operate and maintain information systems in a production environment. includes Post-Implementation and In-Process Reviews.

Disposition
Describes end-of-system activities, emphasis is given to proper preparation of data.

http://didenko.at.ua/publ/common/systems_development_life_cycle/systems_development_life_cycle/64-1-0-149

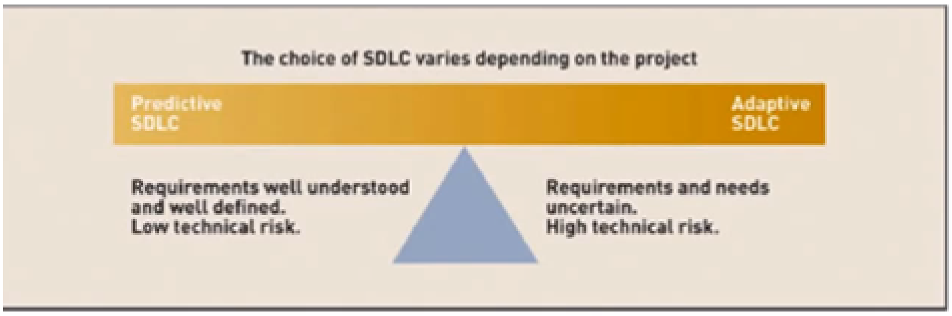
1. **Initiation**: begins when a sponsor identifies a need or an opportunity. Concept Proposal is created.
2. **System Concept Development**: defines the scope or boundary of the concepts. Includes System Boundary Document, Cost-Benefit Analysis, Risk Management Plan and Feasibility Study. You define also the stakeholders and whether the system aligns with the organisational objectives. It is necessary to consider whether it is better to develop the system or procure it (or at least some of its components)
3. **Planning**: develops a Project Management Plan and other planning documents. Provides the basis for acquiring the resources needed to achieve a solution. There are different project management methodologies (PRINCE2 is often used in the healthcare service)
4. **Requirement analysis**: analyses user needs and develops user requirements. Create a detailed Functional Requirements Document. There are a variety of tools and methodologies to transform user needs into user requirements, such as use of stories (storyboarding)
5. **Design**: Transform detailed requirements into complete, detailed System Design Document. Focuses on how to deliver the required functionality
6. **Development**: Converts a design into a complete information system. Includes acquiring and installing systems environment; creating and testing databases; preparing test case procedure; preparing test files, coding, compiling, refining programs; performing test readiness review and procurement activities. Both hardware and software are selected at this stage.
7. **Integration and Test**: demonstrates that developed system conforms to requirements as specified in the Functional Requirements Document (user requirements). Conducted by

Quality Assurance staff and users. Produces Test Analysis Reports. Different testing types are available to test different levels and functions of the system: system testing, integration testing, user acceptance testing.

8. **Implementation**: includes implementation preparation, implementation of the system into a production environment and resolution of problems identifies in the Integration and Test Phases
9. **Operations and Mainteinance**: describes tasks to operate and maintain information systems in a production environment. Includes Post-Implementation and In-Process Reviews
10. **Disposition**: describes end-of-system activities, emphasis is given to proper preparation of data

- A system. after production is still dynamic and thus needs to be maintained and updated. This is often carried out thorugh smaller scale projects.
- **SDLC** is not in itself a system development methodology, but a **series of steps for software and system development**.
- Many SDLC models have been created (e.g. waterfall, fountains, spiral, rapid prototyping, ...)

SDLC Models


The choice of SDLC varies depending on the project

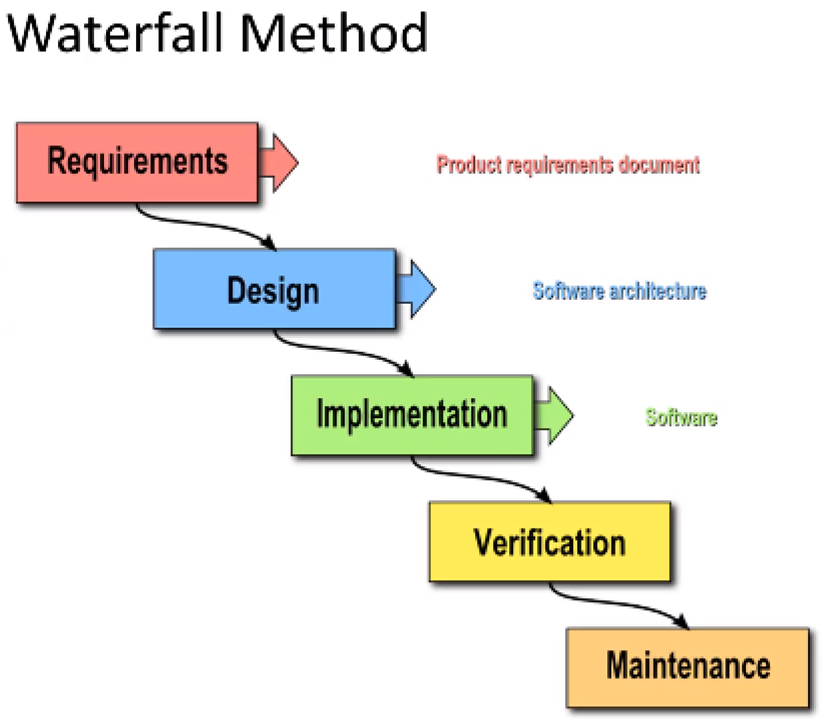
Requirements well understood and well defined. Low technical risk.

Requirements and needs uncertain. High technical risk.

- We can consider a continuum between 2 types of appraoches:
  - **Predictive SDLC**: the development of the project can be planned and organised. This is suitable for **well defined and understood systems** and there is a **low technical risk** (e.g. transferring an old system to a web-based one, with the incorporation of a smartphone app: th e staff already understands the requirements, no new processes are anticipated, the technologies are changing). The predictive approaches were conceived in the 1970s through to the 1990s
  - **Adaptive SDLC**: it is used when the **system requirements are less understood** and therefore the project cannot be planned completely. Some of the system requiremetns would need to be determined after the project started.**Developers** need to be **flexible** and **adaptable** to the project as it progresses. It is only recently in the 21st century that more adaptve approaches have been able to gain ground.

- The choice of SDLC method depends on the type of project considered. The different approaches are not mutually exclusive. Many projects use a **hybrid approach**.

SDLC predictive approaches



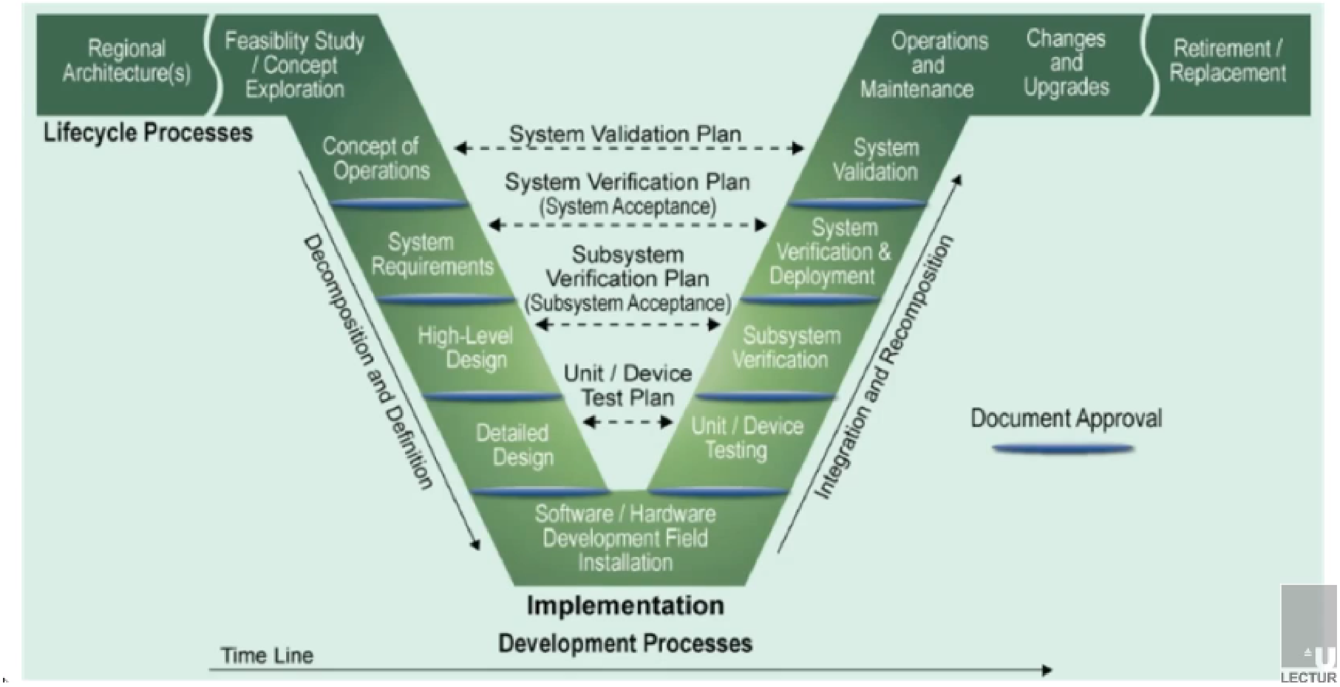
- A traditional predictive approach consists of groups of activities that happen in a linear fashion
- The plot above refers to the **Waterfall Model approach,** the most predictive one in the SDLC spectrum. This model assumes that the phases are carried out **sequentially**:

  1. A detailed system plan is developed
  2. The system requirements are thoroughly specified
  3. The system is designed in detail and then it is programmed and tested
  4. Then it is installed

  In this model there are no phases re-iterations and requires each step to be detailed

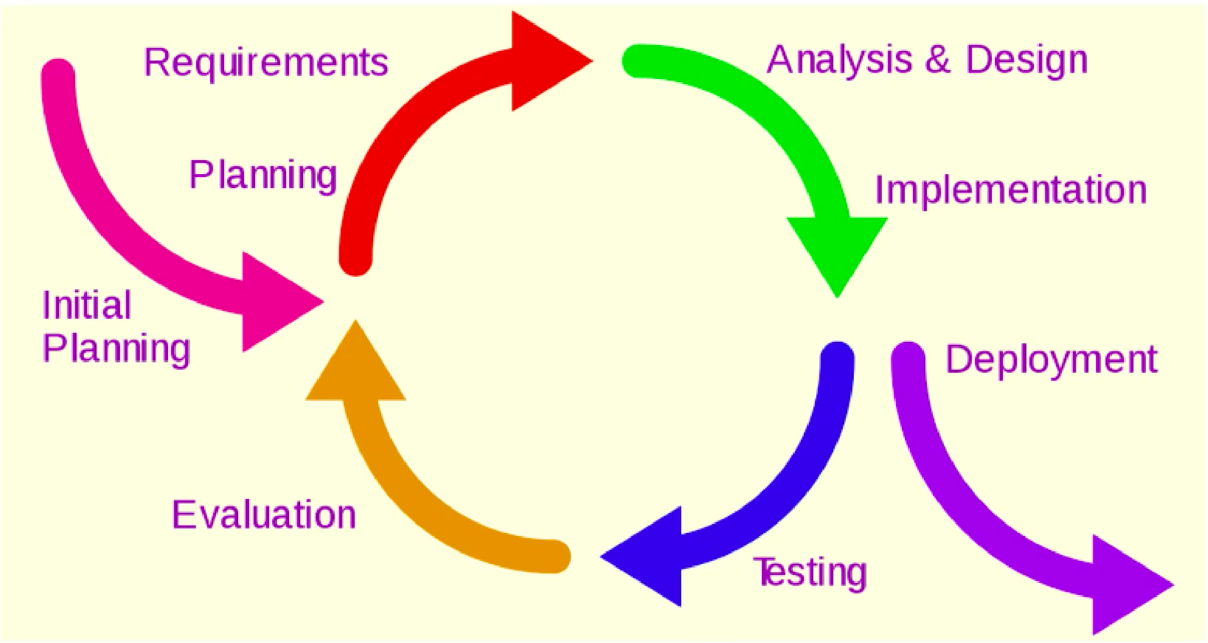
- A slightly different model has been generated as an extension of the Waterfall Model: the **V-Model**

V-model



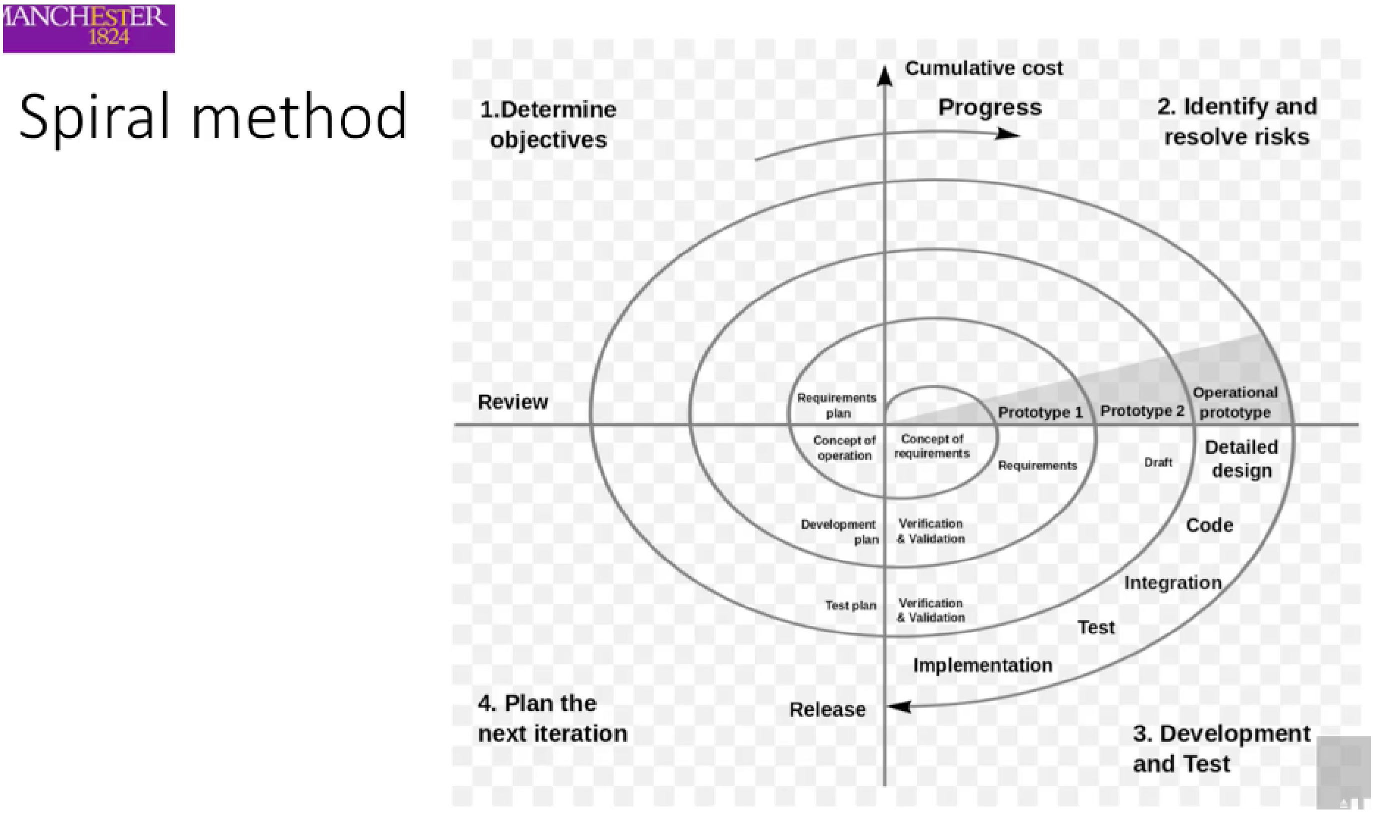
- Unlike the Waterfall model, the V-model is not linear:
  - the first downward phase is the "**Decomposition and Definition**"
  - then, once you have installed the hardware and implemented the system, the second phase is the "**Integration and Recomposition**"
- The V-model allows you to link each stage of the downward and upward phases. There are documents for each stage, highlighting the details for each stage.
- In response to the critics of the Waterfall model, the **Incremental Model** was developed:

Incremental



- The Incremental Model breaks the tasks into **smaller components**, each of which can be developed using a formal methodology. This approach is more adaptive than the Waterfall one. The system is created by **buiding a small subset** and then **adding incrementally more functionalities**. Each increment can follow all the stages depicted in the figure above. The iteration through the stages is shorter than the Waterfall method and that the end uses can be evolved from each increment.
- The Incremental method has been criticised because it makes the whole **project management more complex** because of the required **iterations**, perhaps in a shorter time scale. If you can't break a big project down to components small enough, you might not get the full advantage of using this method.

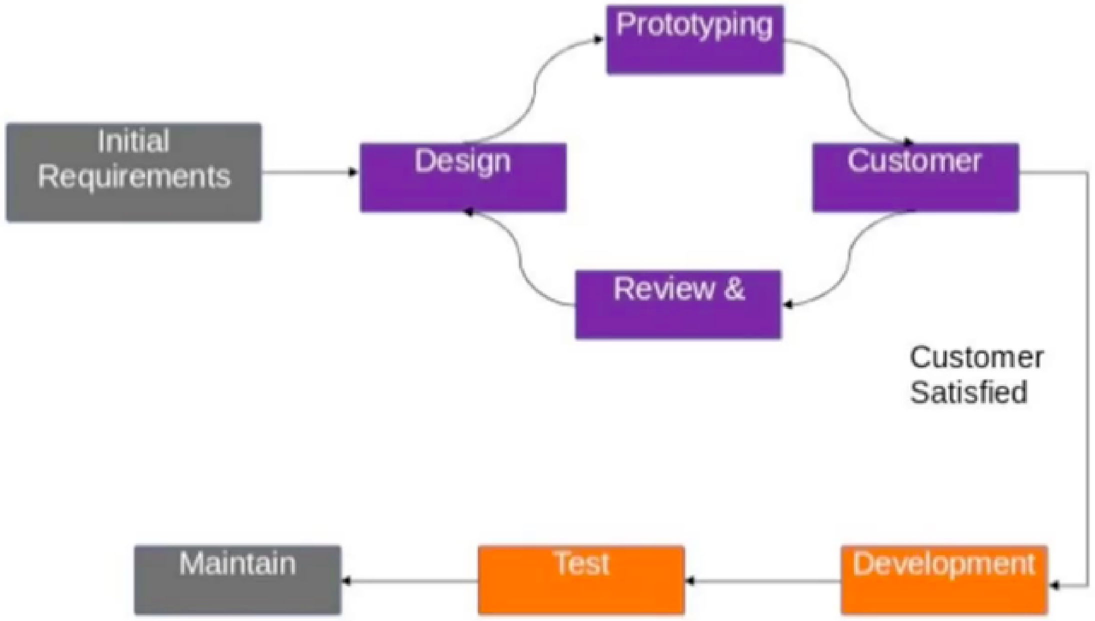
- In general, in an **Adaptive system design approach**, plans and models are adjusted as the project progresses. This type of approach has some of the stages (analysis, design and implementation) overlapping.
- An extreme example of adpative approach is the **Spiral Model**, which contains layers of adaptive elements:



Spiral method

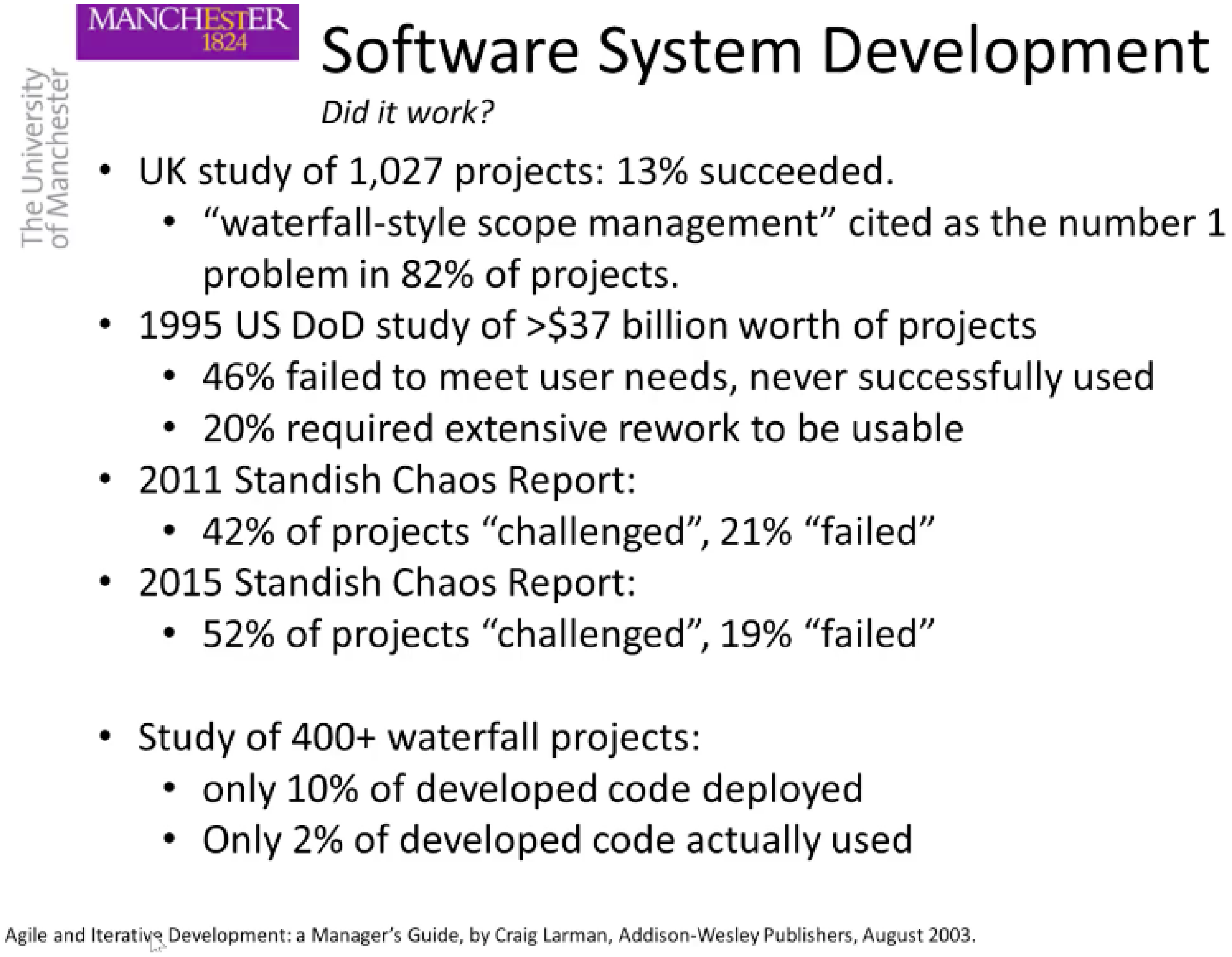
- The Spiral method envisions a project as a **series of stages arranged in a spiral**, which go over and over until the project is completed.
- There are other methods that can be considered adaptive, although not so much as the Sprial method, for example **Prototyping**:

Prototyping



- One of the main problem with other formal methodologies is the **separation between the idea for a system and its implementation**, both in terms of the lenght of time and the agreement (or misunderstaings) between stakeholders. Prototyping allows to get around this quickly, by **creating a prototype without all the complex functionalities of the final product**, enabling **quick testing** of some parts of the design rather than waiting for the whole thing to be ready. It is a proof of concept to testing the desing for appropriateness and it is a great communication tool between developers and end users.
- It is particularly useful to use prototyping for **testing the human/computer interface**, and as an aid for **communications between the technical team and the end users**.
- The main characteristics of prototyping are the following:
    - A **prototype of the system with limited functionalities** is created quickly
    - The prototype allows the **end users to see the system very early** in the process, so they can give their feedback
    - This helps **clarifying user requirements before too much development/effort** has taken place
- **Criticism** of prototyping:
    - Sometimes it might be **wasted effort**, especially if you spent time developing a prototype that doesn't get put into the final system
    - Risk that **features created purely for the prototype are left in the final version** and they do not actually get a full design and analysis (e.g. interfaces to dashboards).

- Producing a prototype too late in the development process might initiate **large**



Software System Development

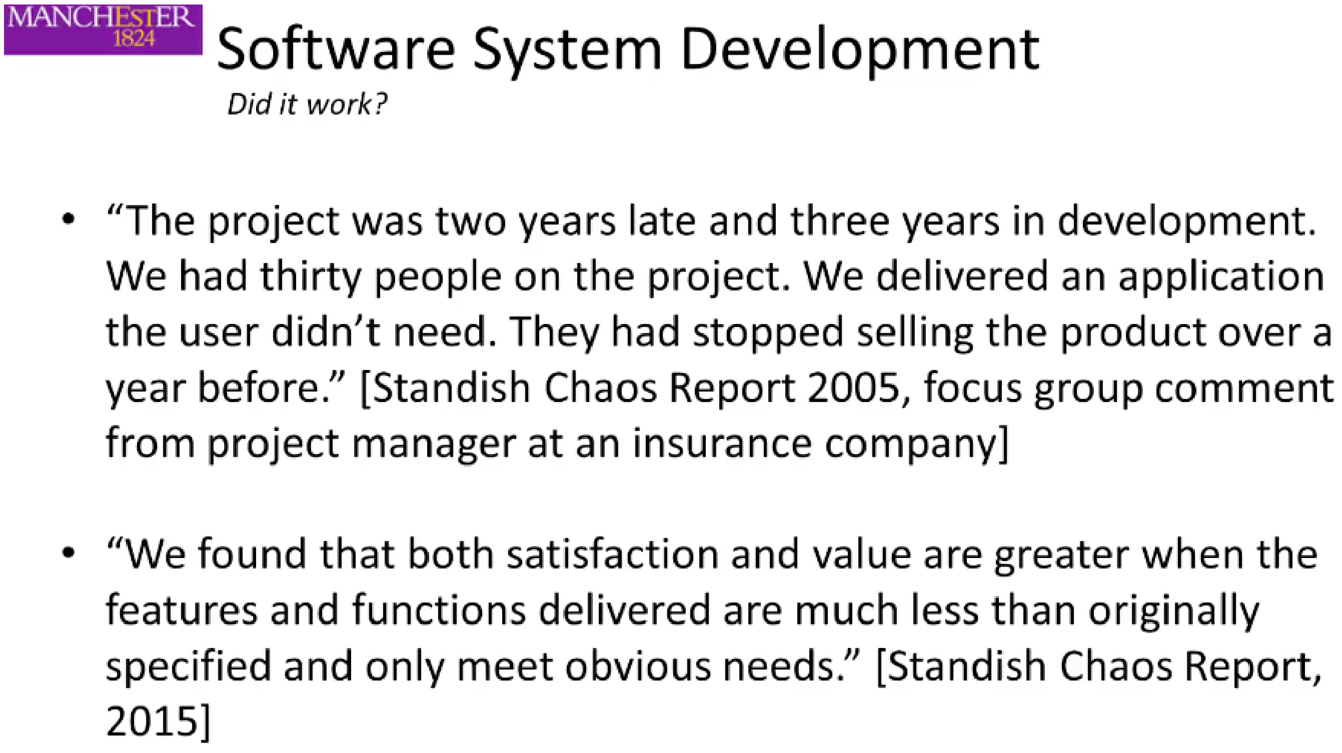
Did it work?

- UK study of 1,027 projects: 13% succeeded.
  - "waterfall-style scope management" cited as the number 1 problem in 82% of projects.
- 1995 US DoD study of >$37 billion worth of projects
  - 46% failed to meet user needs, never successfully used
  - 20% required extensive rework to be usable
- 2011 Standish Chaos Report:
  - 42% of projects "challenged", 21% "failed"
- 2015 Standish Chaos Report:
  - 52% of projects "challenged", 19% "failed"

- Study of 400+ waterfall projects:
  - only 10% of developed code deployed
  - Only 2% of developed code actually used

Agile and Iterative Development: a Manager's Guide, by Craig Larman, Addison-Wesley Publishers, August 2003.

- The problem is that te **Waterfall method** only gets a **"snapshot" of user needs** and it cannot be further adapted.



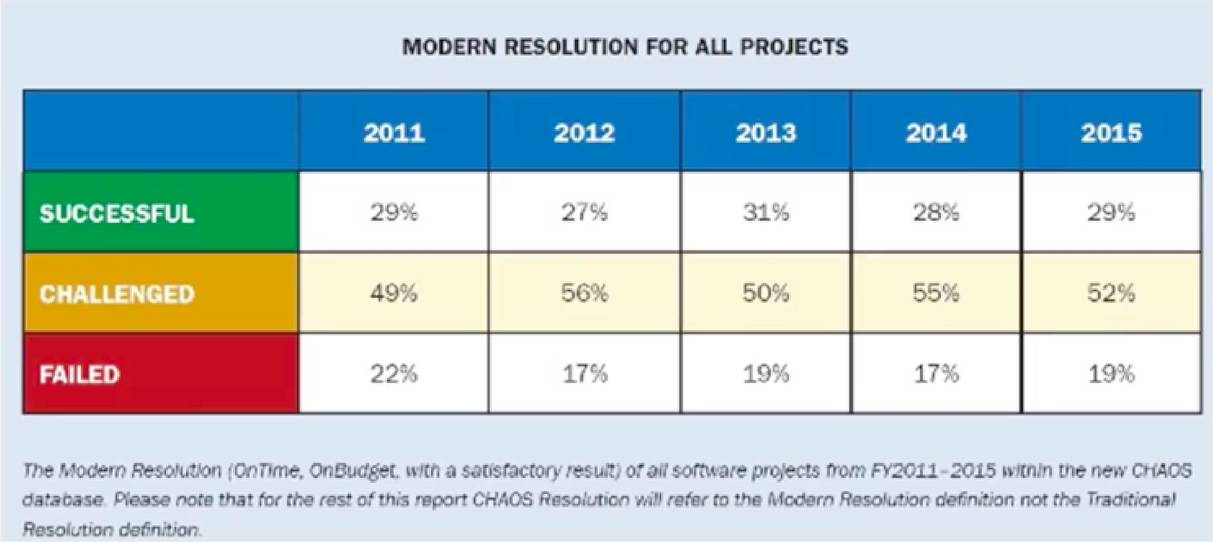
Software System Development

Did it work?

- "The project was two years late and three years in development. We had thirty people on the project. We delivered an application the user didn't need. They had stopped selling the product over a year before." [Standish Chaos Report 2005, focus group comment from project manager at an insurance company]

- "We found that both satisfaction and value are greater when the features and functions delivered are much less than originally specified and only meet obvious needs." [Standish Chaos Report, 2015]

Modern resolution for all projects



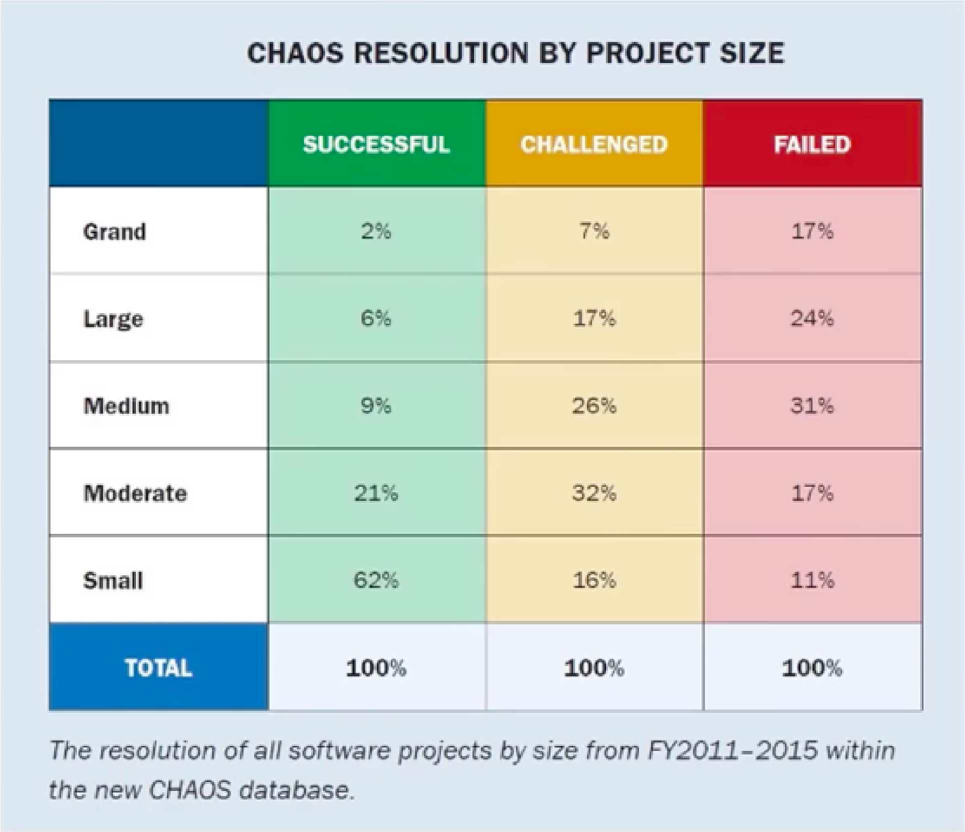
MODERN RESOLUTION FOR ALL PROJECTS

	2011	2012	2013	2014	2015
SUCCESSFUL	29%	27%	31%	28%	29%
CHALLENGED	49%	56%	50%	55%	52%
FAILED	22%	17%	19%	17%	19%

The Modern Resolution (OnTime, OnBudget, with a satisfactory result) of all software projects from FY2011–2015 within the new CHAOS database. Please note that for the rest of this report CHAOS Resolution will refer to the Modern Resolution definition not the Traditional Resolution definition.

- The pattern previously described has not really changed in years.

Small projects have a higher likelihood of success



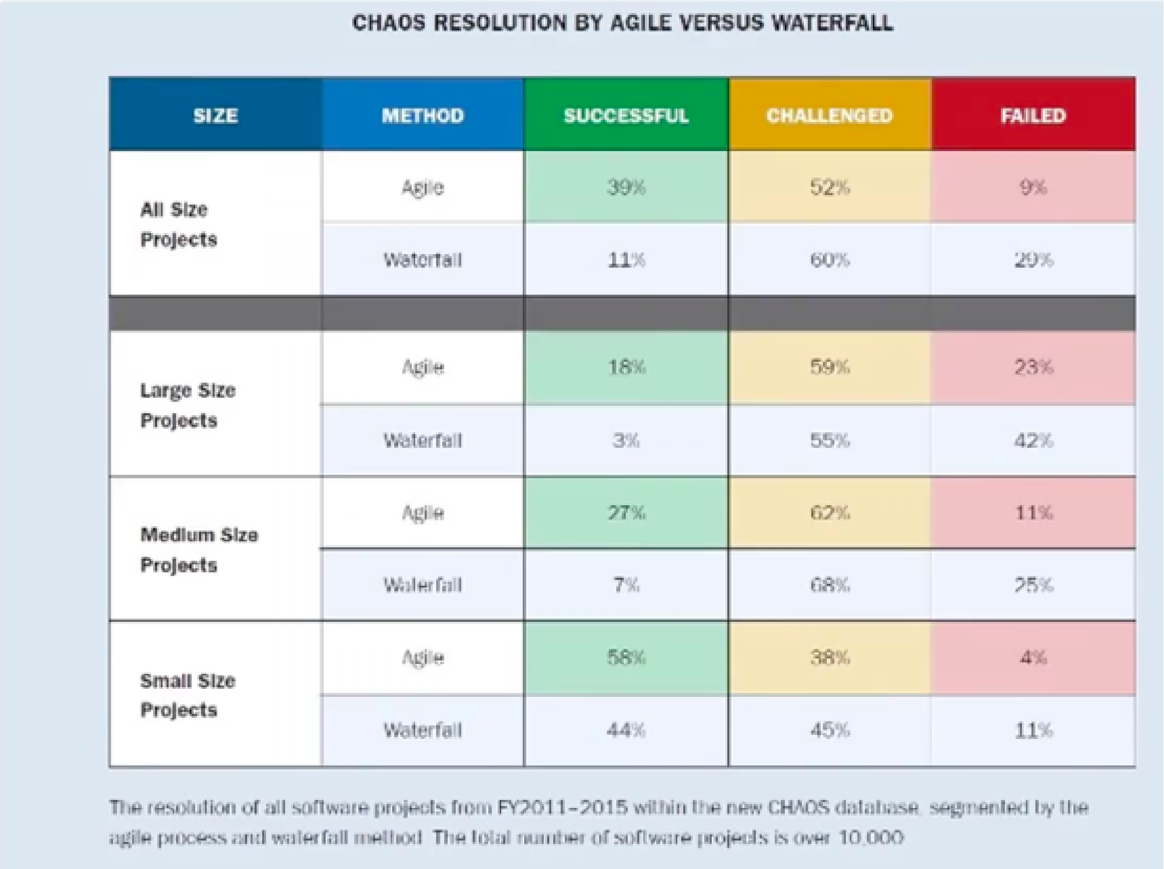
CHAOS RESOLUTION BY PROJECT SIZE

	SUCCESSFUL	CHALLENGED	FAILED
Grand	2%	7%	17%
Large	6%	17%	24%
Medium	9%	26%	31%
Moderate	21%	32%	17%
Small	62%	16%	11%
TOTAL	100%	100%	100%

The resolution of all software projects by size from FY2011–2015 within the new CHAOS database.

- This data suggests that if a larger project could be split into smaller ones it would have a higher chance to be successfull.

Agile versus waterfall

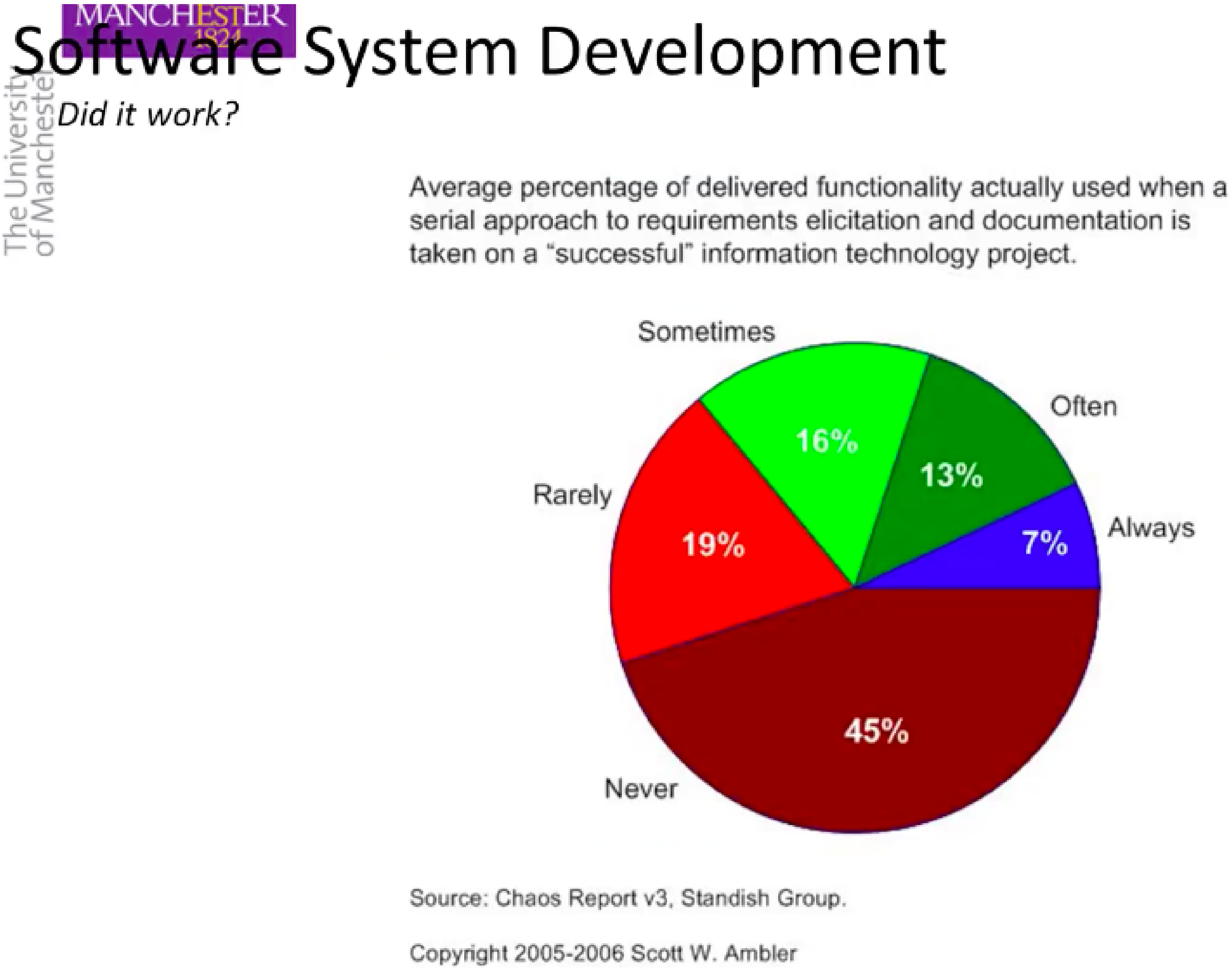


CHAOS RESOLUTION BY AGILE VERSUS WATERFALL

SIZE	METHOD	SUCCESSFUL	CHALLENGED	FAILED
All Size Projects	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Large Size Projects	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Medium Size Projects	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Small Size Projects	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

The resolution of all software projects from FY2011–2015 within the new CHAOS database, segmented by the agile process and waterfall method. The total number of software projects is over 10,000.

- The Agile is the most favoured method for running successful projects.



MANCHESTER 1824

The University of Manchester

Software System Development

Did it work?

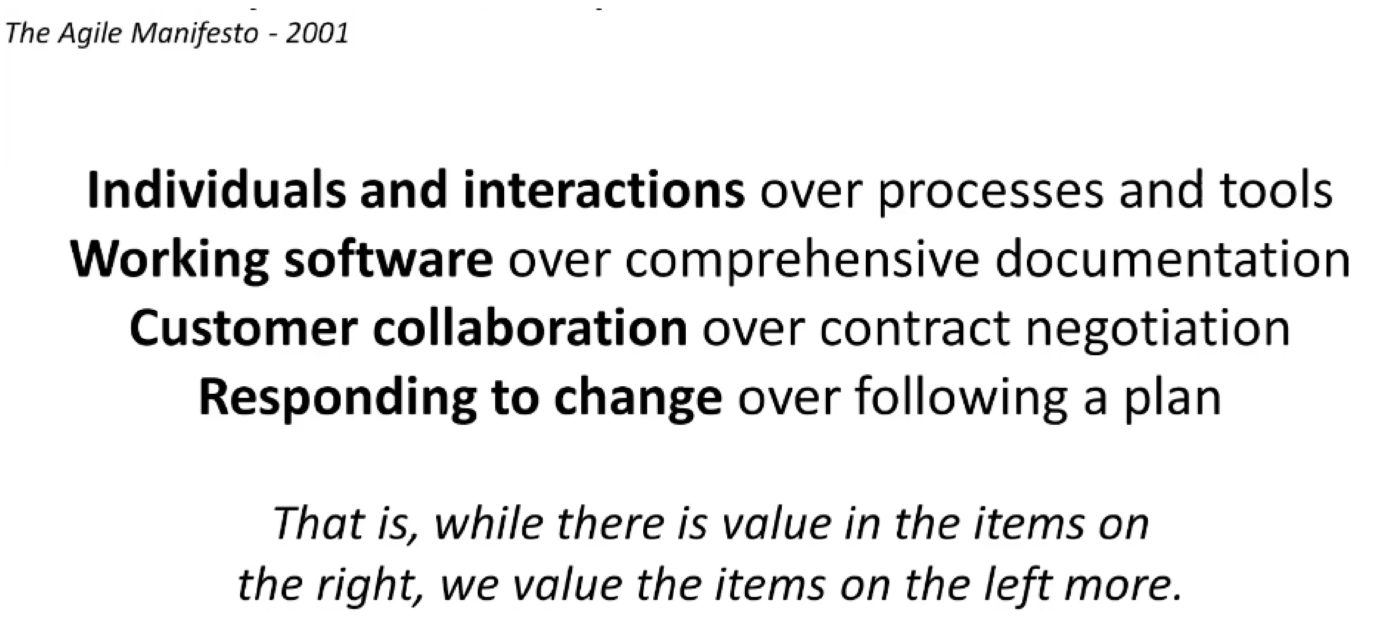
Average percentage of delivered functionality actually used when a serial approach to requirements elicitation and documentation is taken on a "successful" information technology project.

Source: Chaos Report v3, Standish Group.

Copyright 2005-2006 Scott W. Ambler

Agile Method for System Development

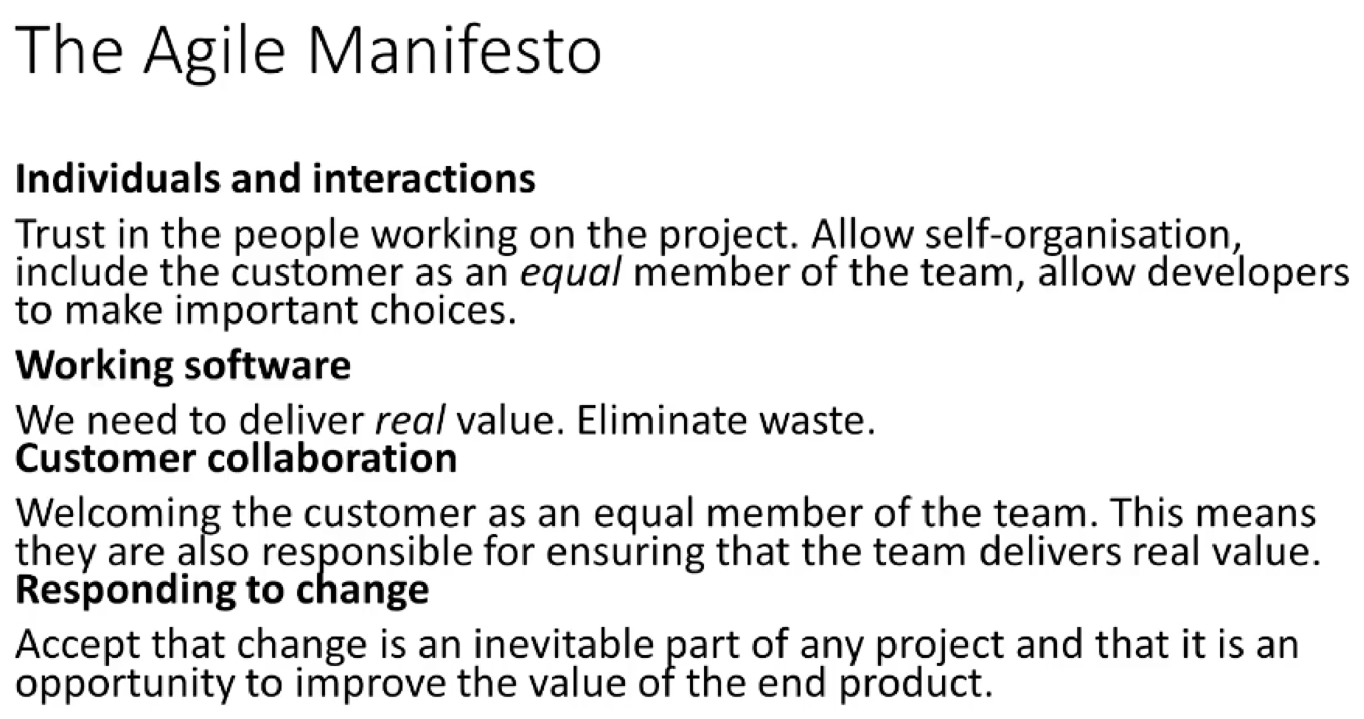
- The most popular approach for system development nowadays is the Agile one.
- Its philosophy and set of guidelines has been thought for a **rapid changing environment**, and **complements the adaptive methodologies** used for system development.
- However it is not often practiced in the NHS or in other healthcare organisations.
- Its manifesto has been signed by many countries and outlines 12 principles + 4 core values



The Agile Manifesto - 2001

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.



The Agile Manifesto

Individuals and interactions

Trust in the people working on the project. Allow self-organisation, include the customer as an *equal* member of the team, allow developers to make important choices.

Working software

We need to deliver *real* value. Eliminate waste.

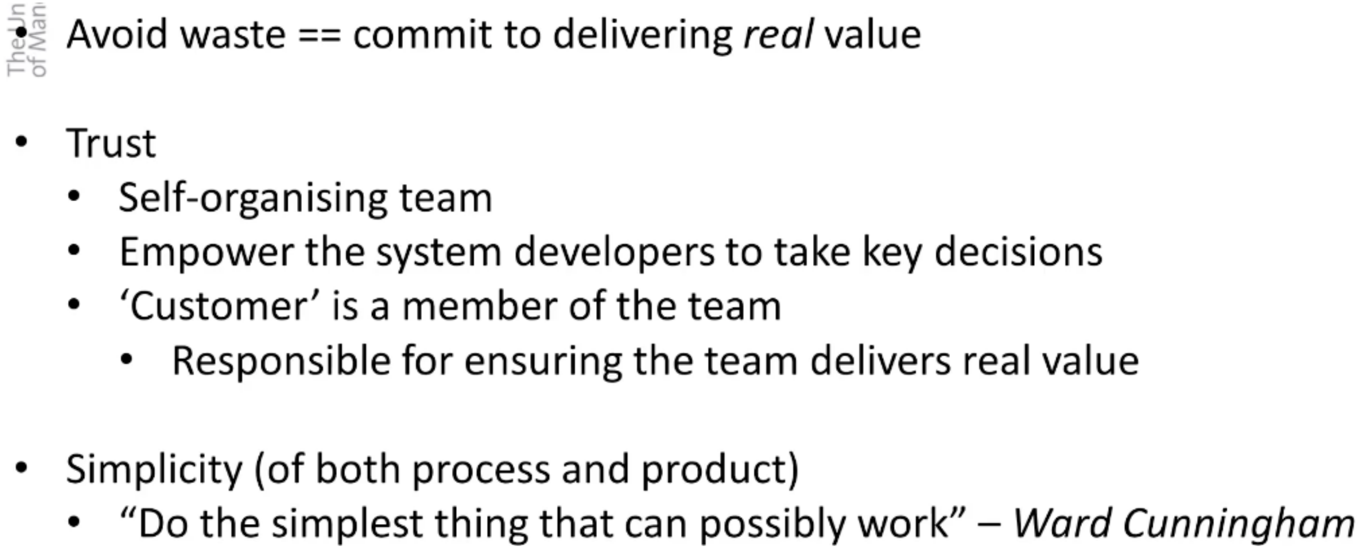
Customer collaboration

Welcoming the customer as an equal member of the team. This means they are also responsible for ensuring that the team delivers real value.

Responding to change

Accept that change is an inevitable part of any project and that it is an opportunity to improve the value of the end product.

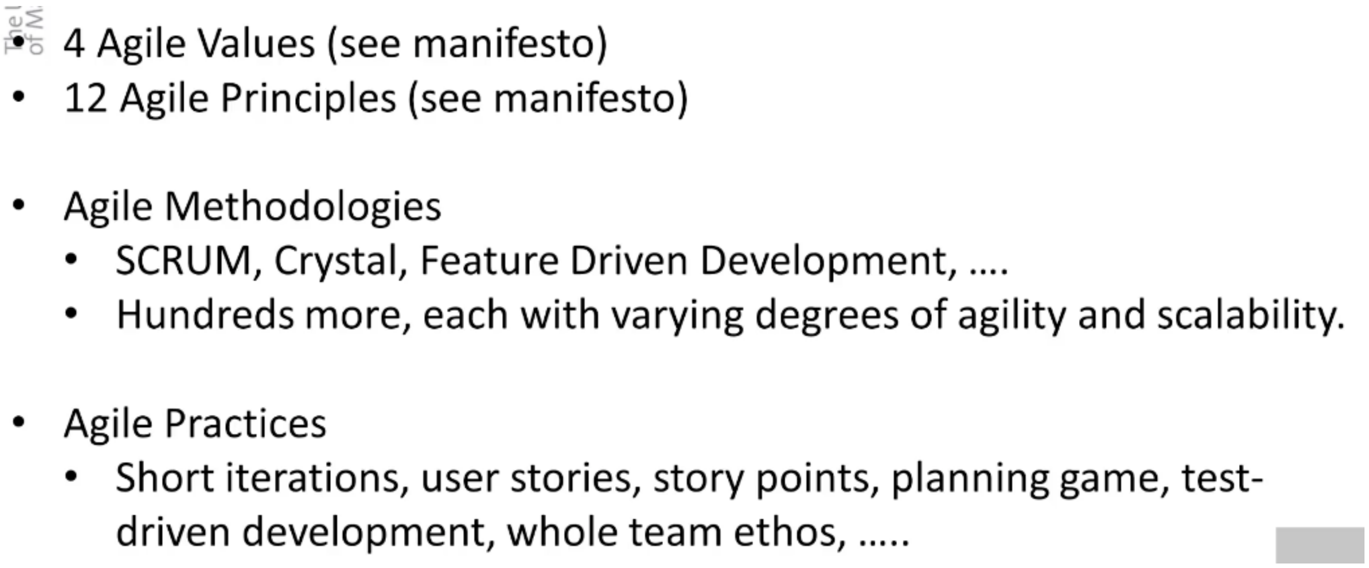
Key ideas for Agile Approaches



- Avoid waste == commit to delivering *real* value

- Trust
  - Self-organising team
  - Empower the system developers to take key decisions
  - ‘Customer’ is a member of the team
    - Responsible for ensuring the team delivers real value

- Simplicity (of both process and product)
  - “Do the simplest thing that can possibly work” – *Ward Cunningham*



- 4 Agile Values (see manifesto)
- 12 Agile Principles (see manifesto)

- Agile Methodologies
  - SCRUM, Crystal, Feature Driven Development, ….
  - Hundreds more, each with varying degrees of agility and scalability.

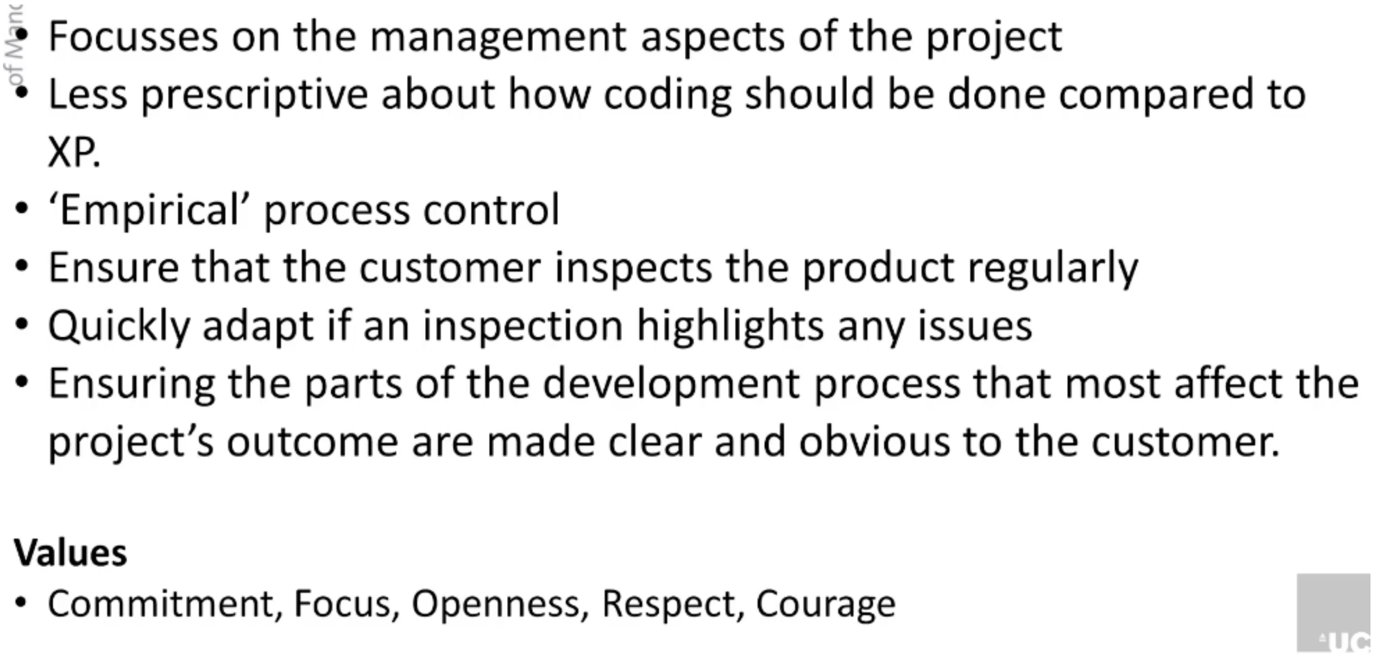
- Agile Practices
  - Short iterations, user stories, story points, planning game, test-driven development, whole team ethos, …..

SCRUM

- The two most used methods use SCRUM and Extreme Processes Programming (XP)

SCRUM

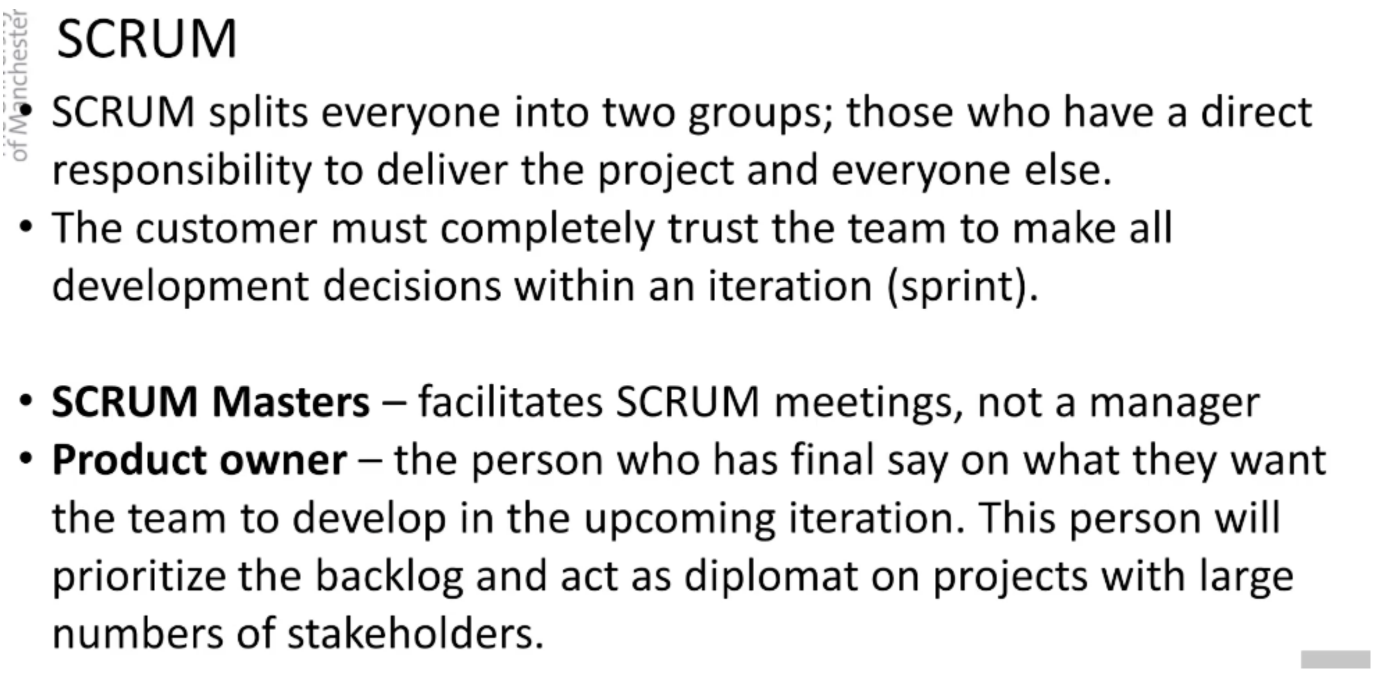
- It is a hands-on system and consists of interlocking steps and components



- Focusses on the management aspects of the project
- Less prescriptive about how coding should be done compared to XP.
- 'Empirical' process control
- Ensure that the customer inspects the product regularly
- Quickly adapt if an inspection highlights any issues
- Ensuring the parts of the development process that most affect the project's outcome are made clear and obvious to the customer.

Values

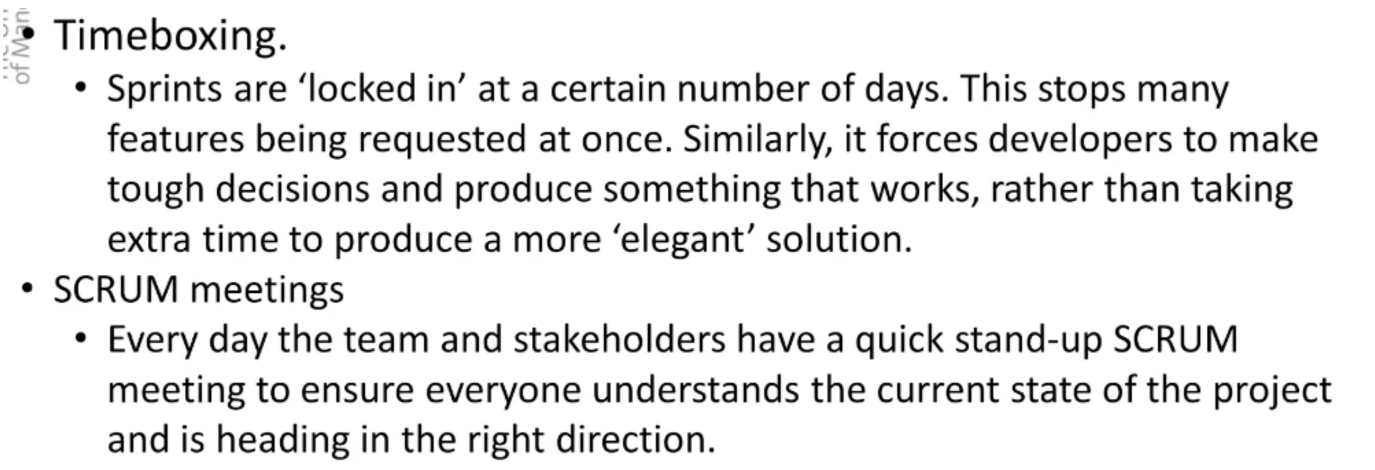
- Commitment, Focus, Openness, Respect, Courage



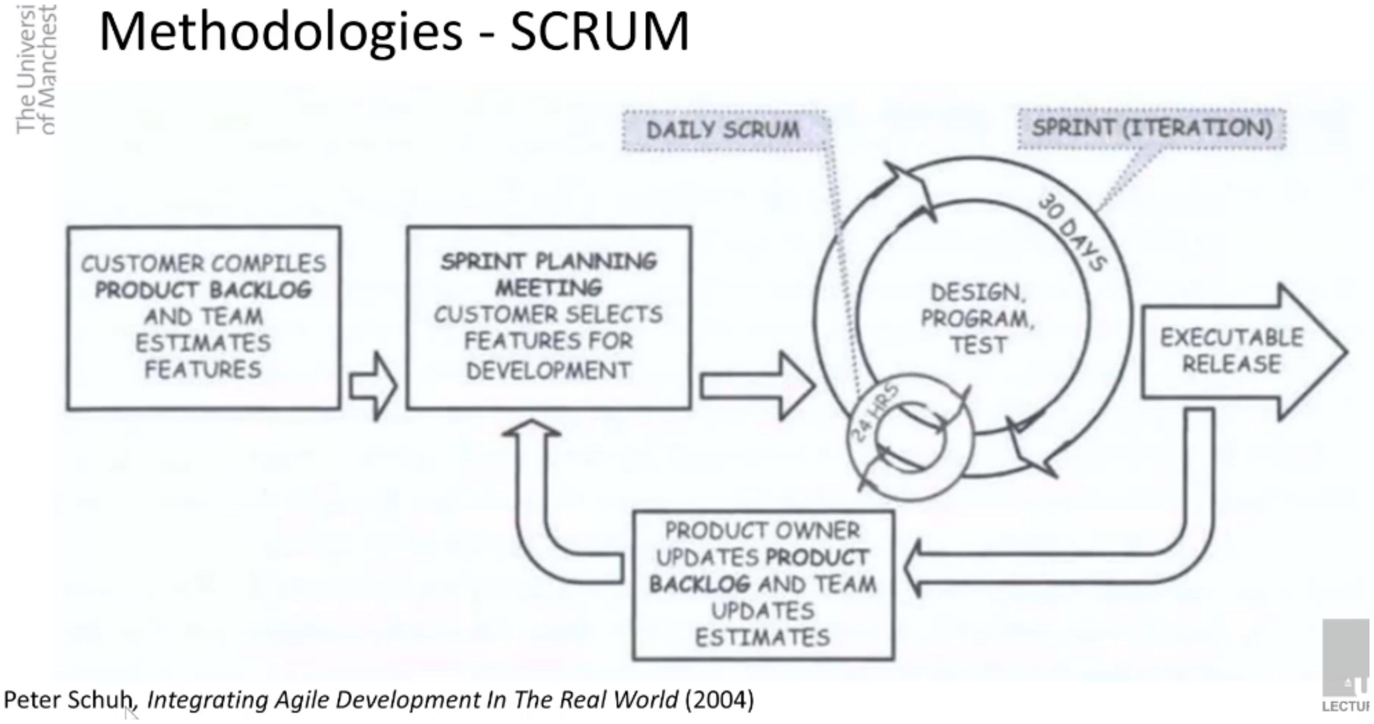
SCRUM

- SCRUM splits everyone into two groups; those who have a direct responsibility to deliver the project and everyone else.
- The customer must completely trust the team to make all development decisions within an iteration (sprint).

- **SCRUM Masters** – facilitates SCRUM meetings, not a manager
- **Product owner** – the person who has final say on what they want the team to develop in the upcoming iteration. This person will prioritize the backlog and act as diplomat on projects with large numbers of stakeholders.



- Timeboxing.
    - Sprints are 'locked in' at a certain number of days. This stops many features being requested at once. Similarly, it forces developers to make tough decisions and produce something that works, rather than taking extra time to produce a more 'elegant' solution.
- SCRUM meetings
    - Every day the team and stakeholders have a quick stand-up SCRUM meeting to ensure everyone understands the current state of the project and is heading in the right direction.



Methodologies - SCRUM

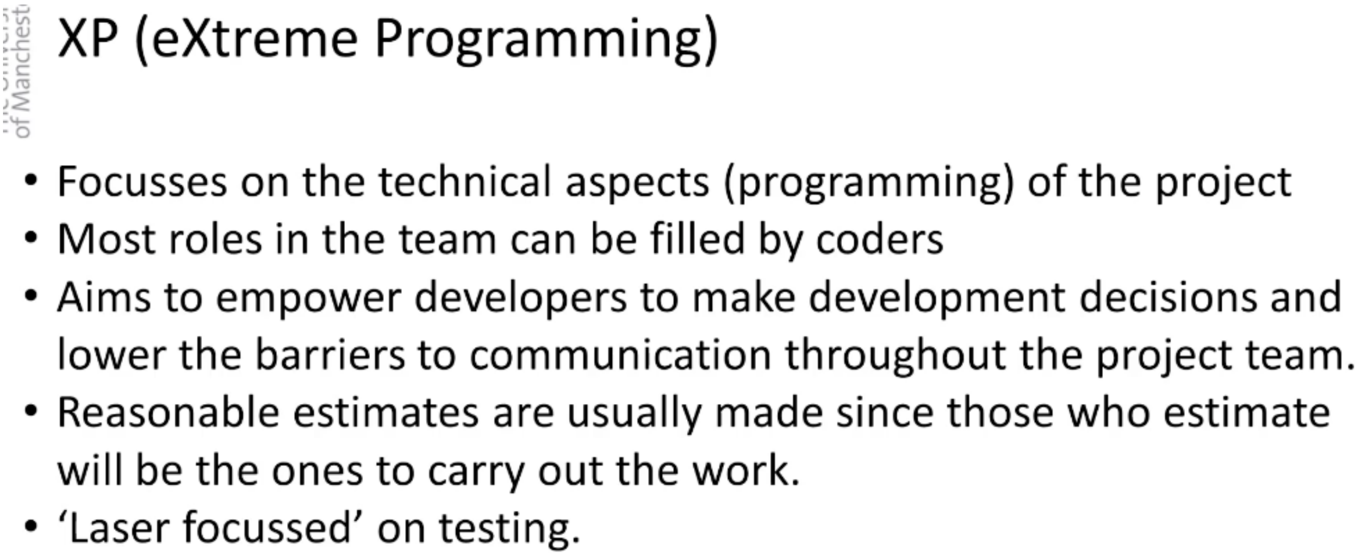
DAILY SCRUM
SPRINT (ITERATION)
CUSTOMER COMPILES PRODUCT BACKLOG AND TEAM ESTIMATES FEATURES
SPRINT PLANNING MEETING CUSTOMER SELECTS FEATURES FOR DEVELOPMENT
30 DAYS
DESIGN, PROGRAM, TEST
24 HRS
EXECUTABLE RELEASE
PRODUCT OWNER UPDATES PRODUCT BACKLOG AND TEAM UPDATES ESTIMATES

Peter Schuh, *Integrating Agile Development In The Real World* (2004)

- Each **sprint** is an **iterative cycle of design/program/test**
- Each sprint can last a variable amoount of time, from 24h to 30 days

XP (eXtreme Programming)

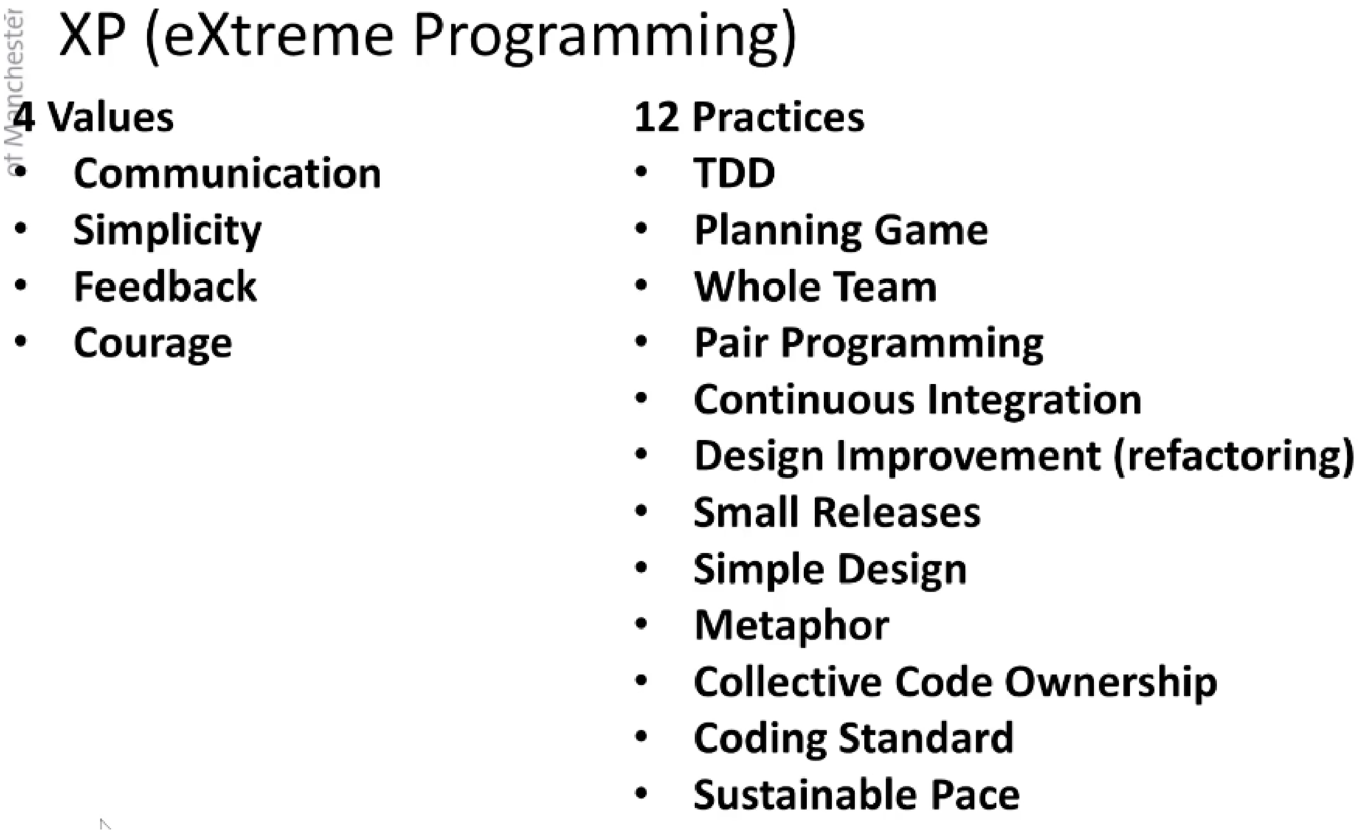
It includes four values, communication, simplicity, feedback and courage (Jeffries, 2001), to combine to 4 basic activities of coding, testing, listening and debugging, defined back in 1999



XP (eXtreme Programming)

- Focusses on the technical aspects (programming) of the project
- Most roles in the team can be filled by coders
- Aims to empower developers to make development decisions and lower the barriers to communication throughout the project team.
- Reasonable estimates are usually made since those who estimate will be the ones to carry out the work.
- ‘Laser focussed’ on testing.

- XP is often used with Agile, although it has also been used with UML and feature-driven development



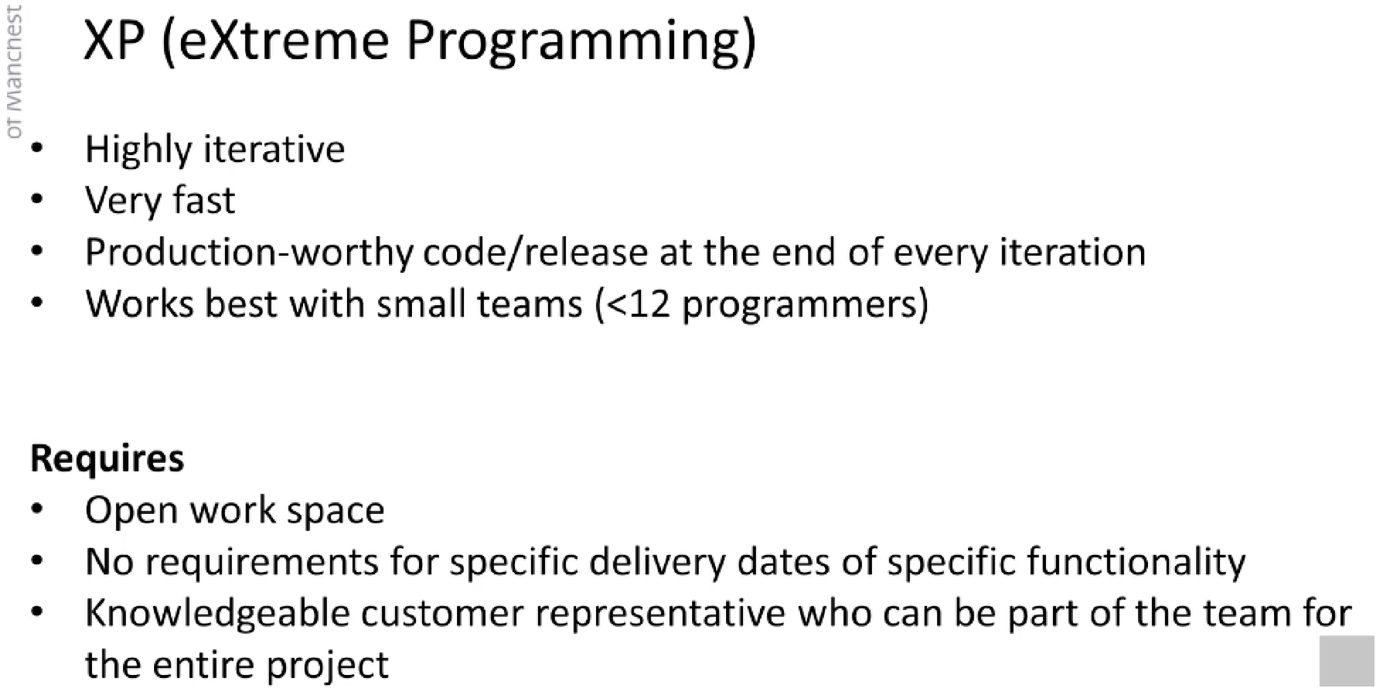
XP (eXtreme Programming)

4 Values

- Communication
- Simplicity
- Feedback
- Courage

12 Practices

- TDD
- Planning Game
- Whole Team
- Pair Programming
- Continuous Integration
- Design Improvement (refactoring)
- Small Releases
- Simple Design
- Metaphor
- Collective Code Ownership
- Coding Standard
- Sustainable Pace



XP (eXtreme Programming)

- Highly iterative
- Very fast
- Production-worthy code/release at the end of every iteration
- Works best with small teams (<12 programmers)

Requires

- Open work space
- No requirements for specific delivery dates of specific functionality
- Knowledgeable customer representative who can be part of the team for the entire project

1.3 Procurement in the NHS

This section provides an introduction to procuring health information systems in the NHS

- Any large organisation will have **rules that govern how it purchases goods and services**. These are designed to help maximise the value and get the best price. This is equally important for government-funded organisations, such as the NHS, in order to guarantee both **effectiveness** and **value for money** for the taxpayer. Additionally, there are specific rules for government organisations and how they purchase in order to ensure **fair competition**, **transparency** and an **equal chance for companies** to compete for business.

- If the **costs of a contract are likely to exceed a specific threshold**, it needs to be **advertised in the Official Journal for the European Union** (**OJEU**). This threshold is currently £189,330 (as of January 2020), so any contract worth more than that (for the lifetime of the contract) must go through the formal process with OJEU. Since this process can be lengthy and challenging, an alternative is to use a **framework**. In the UK, for example, the Crown Commercial Service has effectively already tendered for fixed prices for specific services and so these can be purchased without a full procurement process, making selection of a supplier easier.
- If a contract is less than the current OJEU threshold, the purchasing organisation will need to follow its own regulations while still meeting the requirements for transparency and fairness.
- Now watch the interview with Paul Charnley (under 'Further resources'). Paul, a highly experienced NHS CIO, talks about the procurement process for IT in the NHS, and some of the challenges and opportunities that it creates. As you listen, consider what kind of impact these processes might have, for example on the speed of innovation.
- Also read the information on the NHS webpage on best practice and guidance for procurement of IT solutions within the Digital First Primary Care programme (see reading list).
- After listening to the interview and reading the website, think about what type of requirements for digital procurement might be unique to the NHS and other, similar health organisations? Post your thoughts on the forum.
- For those who are interested in learning more about procurement in the NHS, we have included the 'Procurement guide for commissioners of NHS-funded services' in the reading list. Reading this guide is optional.

Back to summary

1.4 Free or Open Source Software

- Traditionally, open source systems have been driven by academic communities. However, there are more and more healthcare organisations that are taking this approach on, including the NHS. This section gives you a brief introduction into open source programmes relevant to health information systems.
- The NHS now has an open source programme to promote this approach by providing a platform for healthcare professionals and IT suppliers to come together. Another example in practice is the open source enterprise electronic medical record system (openMRS)

platform. To get an impression of the purpose and approaches of open source initiatives, have a look at the websites of these two exemplars.

- One advantage of open source is that it reduces the strangle hold that private vendors have over healthcare providers, and sometimes even over healthcare systems. For example, in the UK, health information systems for primary care are currently provided by two main vendors. This means they can substantially influence what data flows through the system and how.
- Have a look at the short video on open source software (under 'Further resources'). It explains in a fun way what open source is and what the potential advantages are. If you want to learn more about the benefits of open source, there is a 2011 paper titled 'Open Source, Open Standards, and Health Care Information Systems' in the reading list that may be of interest. Reading this paper is optional.

Back to summary

1.5 Thinking about your module assignment

- It may seem like early days, but the sooner you start thinking about your assessment the better. This task will help you get started.
- In section 5 of the module handbook, you will have seen the following introductory text of the section on your module assignment:

  The Trust for which you work has been awarded money from a national funding initiative to either develop, procure or improve a system that supports services for a specific disease area or patient group. You have been asked to work with all relevant stakeholders to produce a structured report that details (a) the current state of play, (b) what innovations your stakeholders require and how the system will address them, and (c) a plan for implementing the system and monitoring its performance. The CIO and CCIOs of your Trust will use your report to advise the Board of Directors on whether this work should go ahead.
- Start with making a list of possible disease areas or patient groups you might investigate. You can choose any area or group you wish, but it is advisable to choose something that is relevant to your professional life. Once you have selected an area or patient group, start searching for trusted information sources that will help you to explore: the scale of the disease area (e.g. incidence and prevalence); its relevance (e.g. impact on patients' lives; societal costs); and the patient pathway(s) involved. Examples of trusted sources are websites and documents produced by the NHS, the National Institute for Health & Clinical Excellence (NICE), (inter)national professional bodies, patient organisations and charities. You could also search for peer-reviewed journal publications.

- Another point to bear in mind is the availability of health information systems for your selected disease area or patient group. To help you think about what system you could use for your assignment, we have provided a list of example systems for cancer, chronic obstructive pulmonary disease (COPD), diabetes and heart failure (under 'Further resources'). Please note there is no obligation for you to pick a system from the table: you can use any system, as long as it is related to your chosen disease area or patient group, is a system that may be developed or procured by a Trust (excluding systems/technologies that, e.g., solely focus on patient self-monitoring without linking or integrating it into care delivery), and allows you to complete the assignment in line with the guidance in the module handbook. You can even choose a system that is fictional, if you prefer, although using a real system will make it easier for you to complete your assignment.
- If you want some early feedback on your ideas for a disease area or health information system for your assignment, post them on the forum with a request for tutor feedback.

Back to summary