good communication throughout

adopting smaller, more easily managed iterations

being responsive to changing customer requirements

the people and how they use it

working software

customer requirements

respond to change

four beliefs

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

Agile

### Rapid Application Development (RAD)

to develop and improve prototypes during each iteration.

uses minimal planning in favour of rapid prototyping. The lack of extensive pre-planning generally allows software to be written much faster, and makes it easier to change requirements.

### Lean

The Lean approach is based on the idea that you should eliminate every goal other than those which create value for the client.

Lean is a paradigm: a relentless pursuit of adding value to customers while eliminating waste and continuously improving the process

Extreme programming

**test-driven development, pair programming, and continuous integration.**

working copies are merged together with a shared “main version” of the code several times per day.

### Velocity Tracking

Velocity tracking is the act of measuring productivity.

counting the number of units of work completed in a certain interval.

Not individual but team

Can be used in both tracking and planning

Each unit of work is chosen by the team – may be hours/days or ideal days or story points.

Backlog – a prioritized list of tasks

Sprint backlog – currently working on

Product owner. Accountable for ensuring that the team delivers value to the business

Authority to accept or reject work.

Responsible for product backlog and its value

Product owner may be part of dev team but not recommended to be scrum master.

Scrum master

Responsible for removing barriers that the team may have

Acts as buffer btwn team and distracting influences.

Keeps team In place (enforcer of rules)

Team

Approx. 7 in size

Team gives estimate of effort to complete each backlog (not the same as value)

Pigs

Commited to the project and scrum process. Scrum master, product owner, team

Chicken

Only gives opinion, not really committed. Stake holders, project managers, and the users.

Scrum Meeting

15mins.

What did you do yesterday, what are you going to do today, what might be a problem. Scrum master resolves the issue.

Release planning

Product backlog is prioritized by product owner

Release dates are not planned in detailed

Defined plan that will give the release a broad direction. Something to aim towards

Definition of “done” be clarified.

Sprint planning meeting.

Held at beginning of sprint

Turns stories into tasks with the scrum master.

Estimates how long each task will take

Decides what item from the product backlog will be taken into the sprint

Team members choose what to accomplish

Sprint review meeting

Hold a demo of what’s been done during sprint

Deliverable functionality is assessed against sprint goals to judge objective met

Pigs and chickens may be involved

Incomplete tasks will be moved to the product backlog

Incomplete work will not be demonstrated

Four hour limit

Sprint retrospective

Reflect on past sprint.

What went well and what can be improved in next sprint?

Was sprint velocity correctly palnned?

Does NOT assess individual performance

3 hour limit

Product backlog

Prioritized list of shit to work. Product owner responsible

DEEP

Detailed appropriately, estimated, emergent, prioritized.

D – user stories in product backlog are sufficiently understood and written for those that need to be completed In the upcoming sprint.

E – value of each item is estimated. Not as accurate as the items at the top

E – not static, changes over time. Stories will be added modified removed

P – prioritized. Valueable on top, not least on bottom.

SDL

**Business Need**

Identify needs of organisation. To do something new, the improve or update something. Very high level and general

**Analysis**

Detail business analysis to identify requirements of system, assess risks & constraints – Output a document, often a detailed Business case

**Requirement Specification -** use UML and Use cases to model the requirements

**Design**

Produce Entity Relationship Diagrams to document database design

Consult infrastructure & environment to ensure there will be sufficient resources to support the new system

**Development**

Coding and testing of modules developed and production of documentation. Unit or component testing often done by developers. Testers may also do some functional testing.

Development environment.

**Testing**

Functional system testing of the whole application, performed by testers and users (User Acceptance Testing)

Non functional testing, including testing the performance of the application, in terms of response time and catering for different loads ie number of concurrent users

Accessibility testing – how easy is it to use, will it be suitable for users with restricted sight or IT ability

Integration testing – do all the components work together

Test environment.

**Release**

Application installed in the live environment.

Support Analyst provide new users with help getting used to the new application.

Infrastructure support will perform the installations.

Development & testing teams may be involved initially to help resolve teething problems

**Support & Maintenance**

Provide support to users if you experience issues.

Central point of contact – Service Desk (1st line)

Support Analysts who specialise in a particular application provide 2nd line support, if the Service Desk cannot resolve an issue

Developers will need to update the code to resolve S/W defects and to introduce enhancements – 3rd line support

Testers will need to perform retesting (to ensure fixes resolve a defect) and regression testing (to ensure fixes do not introduce any new defects)

SLAs (Service Level Agreements) are drawn up to agree the level of support needed by the business

Business needs – problem, benefit, enhancement, technical refresh

Analysis – business analysis, scope, feasibility, risk assessment

Requirement Specifications – definition, use cases, uml

Design – Entity relationship diagram, database design, class diagrams, architecture, test cases.

Development – Coding unites, coding components, build software GUI, build database/stored processes, test plans, unit testing, test cases, test environment, documentation

Testing – system, functional, non functional (load, performance, and accessibility), integration testing

Release – live environment

Support/maint – 1st line: help/service desk 2nd line: analysis 3rd line: code. Regression testing

Training –

Types of development model

* Incremental 4 stages:
  + Initiate project
  + Plan
  + Execute
  + Monitor and control