Projects

Project
A temporary endeavour to create a unique product, service or outcome.

Some few key characteristics are:

- Introduces **change** to the organisation
- Is **temporary**, has a defined beginning and end
- **Cross-functional**, cuts across organisational boundaries
- Deals with the unknown
- Is unique
- Can vary in size, number of resources required (ie. people, costs, and time)

Purpose of projects

- Provides strategic alignment of key activities and visibility at the appropriate levels
- Can be used to prioritise activities
- Allows organisations to deliver changes in a structured and formal manner outside of BAU (business as usual)
- Allows for effective and efficient management of an organisation's limited resources (designated amount of people and budget assigned)
- Establishes ownership and accountability
- Provides clarity and an agreement between different stakeholders about what will be achieved, timeline, who is involved and the purpose
- Should deliver a desired outcome by the end of the project

Project Management

Project Management
The planning, delegating, monitoring and controlling of all aspects of a project, as well as motivating those involved to achieve the project objectives within the expected targets for time, costs, quality, scope, benefits and risks.

There can be many values that come as a result of good project management:

- Allows for organization and structure of resources
- Managing risk
- Identifying and clearing issues
- Managing and implementing change
- Retaining and reusing knowledge
- Organisational wide learning from past successes and failures

Project Manager Skills
Project managers are highly skilled knowledge workers and change agents. The take accountability, make project goals their own and use their skills and expertise to help their team. Technically, they decide on the best allocation of resources, and personally, they engage with other people to report, manage and coordinate their efforts.

Project Manager Key Activities

Traditional management style of a project.

- **Planning**
  They define and clarify the project scope, develop the plan and schedule for the project, and ensure everything is set up in order to achieve the aim of the project.
- **Organising**
  They determine the project team structure, and what roles and responsibilities are needed. They also identify which services are required from external parties, and who is needed for a project.
- **Leading**
  Sets the team direction, coordinates activities across different teams, and assigns work as required.
- **Controlling**
  Tracks the project progress, responsible for reporting the project status, and taking action where needed.

Agile Scrum Master Key Activities

Instead of commanding team members, this style is more suited to spreading the work across team members, and has the Scrum Master facilitating their learning.

- Coaches team and facilitates them to deliver on their goals
- Is invested in the program's overall performance
- Can be an expert and assists people with fixing issues where needed
- Allows each team to organise their own tasks, and coordinates their efforts

Project Management Methodologies

Waterfall

Waterfall is the traditional approach that has been used for over 40 years. The requirements of the project are defined at the start, requires prior planning and allocation of resources. The progress of the project is sequential, moves on from task to task (which are planned beforehand).

Pros: Extensive planning is involved, the thoroughness often results in accurate timelines and budgets (unlikely to go over).

Cons: Difficult to apply changes or modify previous steps, (water can't run backwards), and so needs to be proactive in anticipating problems.

Agile

Focuses on adapting to changing situations rather than using extensive planning. It relies on constant and regular feedback as well as having iterative outcomes, and dividing up work into manageable actions and outcomes. Also has an emphasis on ownership across a team, team members assign themselves work depending on their abilities, and take responsibility for their part.

Pros: Retains flexibility while continually producing outcomes, less rework involved. Also has greater communication and engagement, the team must consistently work together to achieve their goals and bring in their work together.

Cons: Difficult to without experience, esp without an experienced Scrum Master. Can be difficult to contract suppliers, large projects can be a problem, and requires constant communication with client (not always available).

Structured Project Management Methodologies

A process oriented approach that divides projects into multiple stages. Used within different fields (consulting, government etc). Detailed and thorough, and must have a clear need, target customer, realistic benefits and thorough cost analysis.

Eg. PRICNE 2 or COVID roadmaps

Pros: Extensive documentation is helpful with corporate planning and tracking.

Cons: Difficult and untimely to adapt changes and apply all of these changes to all the documentation.

Choosing a methodology

All of the options have a place and can be appropriate. It generally depends on the variables of the projects, such as timelines, resources, people etc. For example, an Agile process might suit a project where we have constant contact with the client, and highly skilled people to work on it. However if we need to stick to a pre-planned budget or timeline, then Waterfall might be a better option.

Determining Success/Failure

Dependent on the outcomes and context of the project, and what happens to it afterwards. If a project is delayed or overbudget, it may still be considered a success if it is used for years to come, or brings in a lot of money.

Successful

A project is completed on time and on budget, with all features and functions as initially specified.

Challenged

A project is completed and operational, but is over budget, over the time estimate or offers fewer features and functions that planned.

Failed

A project is cancelled at some point during the development cycle.

We can determine success by a number of factors, which can be weighted depending on importance (See success factors). However context also plays a large part in how successful a project can be.

Project Initialization

Having a business case is the key to setting up a project for success. We need to have a reason for undergoing this project in the first place.

Business Case

Establishes mechanisms to judge whether the project is (and remains) desirable, viable and achievable as a means to support decision making in its initial and continued investment.

- Provides a base for key decision makers to decide if the project should be undertaken in the first place.
- Demonstrates how the project adds value to the organisation
- Has a set of pre-defined standard organisation characteristics (costs, benefits, risk etc.)
- A living document throughout the project that should be reviewed and signed off at key stages.
- Contains: An executive summary, reasons for why this project is required, expected benefits, risks, time, costs etc.

Investment Techniques

Return On Investment (ROI)
Income divided by investment. The higher the ROI% or the higher the ratio of benefits to costs, the better the ROI is. Many organisations may have a required rate of return (11% to 14%).

$$\text{ROI} = \frac{\text{Total Benefits} - \text{Total Costs}}{\text{Total Costs}}$$

Net Present Value (NPV)
A method of calculating the expected net monetary gain or loss from an investment (project) by discounting all future costs and benefits to the present time. The higher the NPV, the more favourable the project is. Projects with a positive NPV should also be considered if financial value is of important to the purpose of the project.

Payback Period: The amount of time it takes a project before the accrued benefits surpass the accrued costs. Alternatively, how much time it takes for an investment to recover its initial cost. Based on tracking the net cash flow across each year to determine the year that net benefits overtake net costs. For IT, generally want this to be short, due to the changing nature of technology.

Rough Order of Magnitude (ROM)
For each project, we can have a *Cone of Uncertainty* for cost estimates. At the beginning of the project, costs can be uncertain and have a large margin for error in estimations, however as the project progresses, the margin of error will reduce (hence the cone).

Project Charter
Used at the planning stage of a project (very beginning). A page summary of the project description, estimated costs, gains, people involved, a timeline, and a description of the intended outcomes or milestones that are aimed to be achieved.

Processes

Defined Process Control
A process with a well defined set of steps. Given the same inputs, a defined process should produce the same output every time.

Empirical Process Control
A process that adapts to change, learns as we progress, uses short development cycles and uses estimates as an indication or estimate.

Software Development Life Cycle (SDLC)
The systems development life cycle (SDLC) is used to describe a process for planning, creating, testing and deploying an information system.

Activities in SDLC:

- Requirements gathering
- System/Architectural Design
- Implementation and Coding phase
- Testing
- Evolution - deployment and maintenance

Formal SDLC

Waterfall

Advantages

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model
- Phases are processed and complete one at a time
- Documentation available at the end of each phase
- Works well for projects where requirements are very well understood and stable

Disadvantages

- Difficult to accommodate change after the process is underway
- One phase must be completed before moving to the next
- Unclear requirements lead to confusion
- Clients approval is in the final stage
- Difficult to integrate risk management due to uncertainty

Incremental Model

Multiwaterfall cycle, the whole requirement is divided into various releases. Same process but repeated over modules.

Advantages

- Each release delivers an operational product
- Less costly to change the scope/requirements
- Customers can give feedback after each build
- Initial product delivery is faster
- Customers can get important functionality early
- Easier to test and debug during smaller iterations

Disadvantages

- More resources may be required
- More management attention is required
- Defining or separating the increments is difficult and often unclear
- Each phase of an iteration is rigid with no overlaps
- Problems may occur at the time fo the final integration (eg. joining separate parts together)

Formal Model Application

Using a formal model makes sense when we have very clear requirements about the product. We expect projects to require little to no change to those requirements once established, and are clearly defined and well documented. The software and architecture of the product is also well known, and projects on a large scale or require a lot of resources can use a formal model to ensure people stick to budgets and deadlines.

Agile SDLC

Agile

A framework based on the iterative development cycle where requires and solutions evolve through collaboration between self-organising and cross-functional teams.

Agile Framework:

- Manifesto
- 12 Key Principles (see Week 2 Slide)
- Kanban
- Scrum

Manifesto
Very generally, we value individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. While we still value some of these ideals, we value others more.

Kanban
Work items are visualised to provide other team members a view of progress and process, from start to finish. Usually labelled 'TO DO', 'DOING' and 'DONE' and separated by user stories.

Scrum
An agile way to manage a project. A project is separated into a number of sprints, all of which should have a product after it is complete. After each sprint, we can reevaluate priorities and decide what to do for further sprints.

Scrum Roles

Product Owner

- Defines the features of the product
- Decides on the release date and content
- Is responsible for the benefits and profitability of the product
- Prioritises features according to the market value
- Adjusts features and priority every iteration as needed
- Accepts or reject work results

Scrum Master

- Represents management to the projects
- Removes road blocks
- Ensures that the team is functional and productive
- Enables close cooperation across all roles
- Shields the team from external inferences
- Is a member and active participant of the Scrum Team

The Team

- Cross functional, have multiple skill sets such as programming, testing and design
- Co-located (physically or virtually)

Scrum Meetings

Scrum Planning Meeting

- Defines how to achieve sprint goal
- Creates sprint backlog from product backlog (based on User Stories)
- Assigns each ticket story points
- Product Owner priority guides the work
- Release plan is created
- High level design is considered
- Defines what to do, and how to do it

Daily Stand Up

- Short meeting (around 15 mins)
- Discusses what each person did yesterday, what they will do today, and any issues in getting work completed

Showcase

- Team presents what it has accomplished
- Shows a demo of new features etc.
- Informal
- Everyone is invited, potentially showing client

Sprint Retrospective

- Looks at what is and isn't working
- Done after each sprint
- Whole team participates
- Discusses what to start doing, stop doing and continue doing

Scrum Artefacts

User Stories

- A requirement expressed from the perspective of an end-user/customer of the system
- User stories shift the focus from writing about requirements to talking about them
- User stories are short, simple descriptions of a feature told from the perspective of the customer who wants the new capability of the system.
- Pattern: As a {type of user} I want {some goal} so that {some reason}

Story Points

- Used as a unit of measure for expressing an estimate of the overall effort that will be required to fully implement a product backlog item or any other piece of work.
- Helps estimate how much work can be done in a sprint.

Sprint Backlog

- Seldom altered
- Contains all the user stories that should be developed over the sprints
- Can be decomposed into smaller stories

Burn Down Chart

- A graphical representation of work left to do versus time
- Outstanding work is on the vertical axis, time is on horizontal
- Used to predict when all of the work will be completed

Advantages

- Customer satisfaction by rapid, continuous delivery of usable software
- People and interactions are emphasised rather than process and tools
- Continuous attention to technical excellence, good design and quality
- Regular adaptation to changing circumstances

Disadvantages

- Difficult to assess the effort required at the beginning
- Can be very demanding (from traditional approaches) on users times
- Harder for new starters to integrate into the team
- Agile is a very different approach - It can be intense for the team
- Requires experience resources (which are limited in today's market)

Risk Management

Risk
An uncertain event or condition that, if it occurs, has a positive or negative effect on the project objectives. If it has a positive impact, it can be considered an *opportunity*

Uncertainty
Lack of complete certainty about an event or outcome. A risk is the result of uncertainty, but not every uncertainty is a risk (or relevant to our project).

Risk Management Process
A circular process, as risks can constantly change, and need to be updated. As a result, we should always be considering these actions in a project.

- **Plan:** How to approach and plan risk management activities?
- **Identify:** Identify the possible risks
- **Analyze:** Identify the relative priorities of the identified risks (Both qualitative and quantitative)
- **Action:** How can we reduce the likelihood or impact of risks?
- **Monitor and Control:** How can be detect the ongoing status of our risks? How can we control them effectively and efficiently?

Risk Management Plan (RMP)
Documents the procedures for managing risks throughout a project. The project team should review the PMP to understand and implement the approaches to risk management.
A typical risk management plan includes:

- Methodology
- Roles and responsibilities
- Budget and schedule
- Risk categories
- Risk probability and impact
- Tracking
- Risk documentation
- Contingency plans
- Fall back plans

Identifying Risks

To determine if an event should be considered a risk, we can ask the following questions. Is the probability of the event occurring greater than zero? What is the impact of the event on the project? Do we have any degree of control over the event or the outcome?

Generic Risks
Risks common to every software project, eg. budget.

Product-Specific Risks
Threats or opportunities specific to the product, and can only be identified by people who have a good understanding of the product and technology. **Project Risks**
Affects the planning of the project.

Product Risks
Affects the quality or performance of the project outcome, or its development process.

Business Risks
Affects the economic success of the project. Happens after the project's development and tends to be for the company/stakeholders rather than developers.

Risk Identification

Deals with using a systematic approach for identifying and creating a list of threats and opportunities that may impact the project's goals. **Pondering**
Involves an individual sitting and thinking about the possible risks that could occur in the project. This is one of the initial risk assessment tasks used in many projects.

Interviews/Questionnaires
Involves interviewing relevant project stakeholders, or asking them to fill our a questionnaire to use their knowledge of a domain. As its unlikely that a risk manager in a software project would be aware of all the risks relevant to the domain, input from stakeholders and domain experts is essential.

Brainstorming
The team can use a risk framework, or the Work Breakdown Structure to identify threats and opportunities. The point of this method is to have everyone contribute, then discuss and evaluate the risks together.

Checklists
Uses standard checklists of possible risk drivers collated from prior experience or projects. These are then used as triggers for experts to think about the possible types of risks in that area, and apply it to their own project.

Delphi Technique
A group of experts are asked to identify risks and their impact (probability and scale). They then share the results anonymously and ask the experts to update their responses based on other people until a consensus is reached.

SWOT Analysis
Strengths, Weaknesses, Opportunities and Threats. Allows finding strengths and weaknesses as well as risks.

Risk Analysis

Identifies each risk's probability and impact. Can be qualitative (done based on experience or intuition) or quantitative (based on maths/stats - not covered in this course)

Risk Assessment

Prioritises risks so that an effective risk strategy can be formulated.

Qualitative Risk Analysis

1. **Estimate the risk probability (P)**
   What is the probability that the risk will occur? Usually a value is assigned based on expert judgement, such as prior execution of said project.

2. **Estimating the risk impact (I)**
   The impact the risk will have on the project. Usually done on a scale (eg. 1-10 where 10 is catastrophic).

3. **Compute the risk exposure (P*I Score)**
   Risk Exposure $= P \times I$

4. **Ranking Risks**
   After the exposure is calculated, we can rank risks in order of importance/likely to happen. This will help decide which risks are considered in decisions/plans etc.

5. **Identify the root cause**
   The root cause of all risks. If this can be identified, then all of the risks that occur can be controlled by addressing the root cause. Think why does this risk occur, and how can we prevent it?

Risk Matrix

Defines the level of risk by considering the probability or likelihood consequence severity. It visually represents the likelihood/degree of risks and assists management in decision making. **Risk Response Strategies**

While we try to identify all relevant risks, not all of them can be addressed as they require resources. Therefore, appropriate response strategies need to be selected for each risk.

Accept or Ignore

If we believe that the risk is of an acceptable exposure level, then we can hope that it won't occur, or that if it occurs then costs of resolving this risk is higher than just letting it occur.

Avoid

We completely prevent the risky event from occurring, by either ensuring its probability is 0 or that the impact is 0.

Mitigate

This involves employing techniques to reduce the probability of the risk, or reduce the impact of the risk. This results in a residual risk, where the risk is from the same event, but has a lower probability or impact. We would then analyse the residual risk as we would the original risk.

Transfer

Involves transferring the burden of the risk to another party. For example, taking out insurance is an example of a risk transfer, where the risk is offset by payments from the insurer.

Opportunity Strategies

Details how to handle an opportunity if it arises from the result of a risk.

Exploit

Add work or change the project to make sure the opportunity occurs.

Enhance

Increase the probability and positive impact of risky events.

Share

Allocate ownership of an opportunity to a third-party.

Accept

This means that we believe that the cost to exploit or enhance is not justifiable so we do nothing about it.

Risk Response Plan

Also known as a Risk Register, one risks and strategies are identified, they can documented. An example template is show below:

Risk ID	Trigger	Owner	Response	Resources Required
1	A key member leaving the project	Project Manager	Hire replacement	Costs of hiring

Monitoring and Controlling Risks

Tools are used to monitor the triggers, and additionally add or remove risks as they arise.

Risk Audits

The external team looks at the comprehensiveness of the identification process and ensures other procedures and processes are in place.

Risk Reviews

Internal reviews of risks periodically that results in status reports generated for PM and those who need-to-know.

Risk Status Meetings

Risks must be reviewed and discussed in project status meetings, which are periodically held in projects.

Agile Risk Management

Identification

Uses a doomsday clock or karma day model to represent risks. Each clock is separated into sections and everyone involved with the project writes a risk on a sticky note and sticks it in the relevant section.

Analysis

A graph of risks and opportunities is created, where the benefits are on the left of the x axis, and the costs are on the right. Additionally, the probability of the risk occurring is on the y-axis, and the sticky notes are placed where appropriate (benefit or cost, low or high probability)

Sprint Review Risk Evaluation

The format of a risk item in the product backlog can vary. We can sometimes use Feature Driven Development (FDD).

[action] the [result][by|for|of|to] a(n) [object]

Project Management

Project Management Plan

A formal approved document that defines how the project is executed, monitored and controlled. It is a document that is owned, controlled and filled in by the Project Manager and is updated and used throughout the project. It is also the Project Manager's job to put together, but they also consult relevant stakeholders.

Project Information

A section included in the PMP.

- Executive Summary
- Financial Authority to proceed
- Key stakeholders
- Scope
- Delivery approach (eg. SDLC)
- Resources and people
- Key Milestones
- Project Budget
- Lessons learned applied to this project
- Constraints

Project Governance

- Roles and Responsibilities
- Mandatory project planning
- Schedule
- Risk Management
- Cost Estimation
- Quality Assurance
- Configuration Management

Project Charter

A summary project proposal to secure approval for the project goals and terms. Mainly used to communicate key information to stakeholders for approval.

Stakeholder Management

Stakeholder Register
Identifies *all* the stakeholders, and their roles in the project. Other information such as their position and contact details may also be included.

Stakeholder Engagement
A degree of involvement for stakeholders, which influence how we interact with them.

- **Unaware:** Unaware of the project and its potential impacts on them
- **Resistant:** Aware of the project and resistant to change, or the project.
- **Neutral:** Aware of the project, neither supportive nor resistant
- **Supportive:** Aware of the project and supportive of change
- **Champion/Leading:** Aware of the project and drives change

Stakeholder Management Plan
Not every detail of a stakeholder is necessary in a PMP, so the SMP is where we plan on how to engage with stakeholders. Includes:

- Current and desired engagement levels
- Interrelationships between stakeholders (eg. resistant stakeholder is friends with supportive one, can convince?)
- Communication requirements (eg. stakeholder like receiving weekly updates etc)
- Potential management strategies for each stakeholder
- Methods for updating the stakeholder management plan

Stakeholder Analysis
Includes:

- Names and organizations of key stakeholders
- Roles in project
- Information about each stakeholder
- Their level of interest
- Their influence (eg. don't want to annoy person funding it)
- Suggestions and strategies for managing relationships with each stakeholder

Communication Management

Project Management
As a person managing a project, you have to be able to convey your own thoughts while listening to others. You need to motivate and influence others, manage conflict and delegate work.

Key Skills

- Communicate with the client
- Run a meeting
- Manage a team
- Influence your environment
- Communicate your thoughts articulately (whether in meeting, or in a report etc.)

Communication Plan
Used by PMs to assist in managing and coordinating key communication messages. It can set a common understanding in what is required, what needs to be done, and ensure that communication for all relevant stakeholders is effective and efficient.

It defines:

- What information will be communicated - detail and format
- Communication channel to be used
- What information is shared, its frequency, and to who
- Communication needs of stakeholders
- Constraints which can affect a project
- Resolving conflicts or issues
- And various other points

Virtual Team
Refers to a group of individuals who work together from different geographic locations and rely on communication technology. Though communication is less rich than face to face, there are some benefits, such as others feeling more comfortable.

Communication Charter
Discipline about how the team should communicate, for example discussing a reasonable time to receive a response. Can include muting oneself when not speaking, using email for formal correspondence etc.

Conflict
Deal with conflict quickly before it becomes a major issue. Usually dealt with by the PM, and can arise from a variety of reasons.

Project Scheduling (Formal)

Project Schedule
An important artefact that is generated during the project planning phase. Is used and maintained throughout the project to monitor and track project progress. Contains the duration and dependencies of each task, the resources and people involved for each task, as well as milestones, deliverables and the project timeline.

Developing a Project Schedule

- Break tasks down into smaller, more manageable chunks (Can use the work breakdown structure WBS)
- Identify the interdependencies between the broken down tasks and develop a task network
- Estimate the effort and the time allocation for each task
- Allocate resources for tasks and validate effort
- Develop the project schedule

Work Breakdown Structure
We split tasks so that they have a specific outcome or deliverable, and then look at all the smaller tasks necessary to achieve that goal. For example, the larger task of decorating a room can compose of purchasing materials, preparing a room etc. The purchasing materials task can be further decomposed into more tasks like buying paint, buying a ladder etc, but they have the common outcome of having the materials necessary. We can also note that we cannot paint the room until the paint is purchased, creating a *dependency*.

Identifying Task Dependencies
There are various types of tasks, and dependencies between them determine the order and timeline of what tasks need to be completed.

Unconstrained Task
If a task has no dependencies and can start at any time.

Constrained Task
When a task depends on another task. This can be caused by a task needing the output/product of another task, or a task needing the resources being used by another task.

If task **B** depends on task **A** then $\mathbf{A} \rightarrow \mathbf{B}$
Where **B** is a successor task (S) and **A** is a predecessor task (P)

Task Types

Dependency	Description
Finish to Start	Predecessor must finish before Successor can start (most common type)
Start to Start	Predecessor must start before Successor can start
Finish to Finish	Predecessor must finish before Successor can finish
Start to Finish	Predecessor must start before Successor can finish

Task Network
A tool used to display dependencies between tasks. If a task is dependent on another, then an arrow is used to denote a dependency. Otherwise, tasks can be represented as parallel (stacked or in similar positions).

Effort Time Estimation

Person Months
Also known as man months, this is a common measure for estimating the effort for software. It is the time in months for a single person working full time to complete the task. (Note that adding a person doesn't mean that the man months will reduce by a month)

Time Estimation
An estimate of the time a project could take.

$$T_E = \frac{(O + 4M + P)}{6}$$

Where O = optimistic time (best possible case), P = pessimistic time (worst possible case), M = most likely time, T_E = expected time
We generally consult experts for these time estimates.

Resource Allocation While computing the number of people required for a project is simple, assigning resources and people for tasks is more involved. This is due to people having different areas of expertise and availability, so other factors like who works well together, or where someone would be best suited also needs to be considered so that project can have the best outcome.

Number of Personnel
If the effort (person-months) and the time is known, the number of people required can be computed as

$$N = \frac{Effort}{Time}$$

Project Schedule
Answers two major questions: How long will the project take, and how much will it cost? The most common notations we use are Gantt charts and PERT charts.

Activity (Task): Is part of a project that requires resources and time.
Milestone: Is the completion of an activity that provides evidence of a deliverable completion, or the end of a phase (should take zero time and just denote completion of a task).
Free float (free slack): Is the amount of time a task can be delayed without causing a delay to subsequent tasks.
Total float (total slack): The amount of time a task can be delayed without delaying project completion.
Critical path: The longest possible continuous path taken from the initial event to the terminal event (A path with no slack).
Critical activity: An activity that has total float equal to zero.
Deliverable: Specific artefacts that are of interest, eg. a prototype, document etc.

Gantt Chart
A bar chart that shows the task schedule against a timeline or calendar. Can be used to view planned activities and deadlines against current progress, and monitor project progression (denoted by shading). Can also show dependencies between tasks.

PERT Chart
An activity network that shows the dependencies among tasks and the critical path. Shows dependencies, slack, critical paths and allows project managers to do scheduling tradeoffs.

ES	Duration	EF
Task Name		
LS	Slack	LF

Where

- ES = Earliest start time
- LS = Latest start time
- EF = Earliest finish time
- LF = Latest finish time

A task can be represented using one of these nodes, and then the dependencies are connected using arrows. We can have multiple critical paths, where critical paths are ones with a total slack of 0 (and also the earliest possible start and finish of dates). A delay in this path means the dentire project will be delayed, however if we can shorten the path then the duration of the project can also be shortened. This can be done by removing dependencies from the critical path, or reducing the activities duration.

Project Tracking and Control
There are various methods of tracking progress, from having regular meetings or reviews of progress and milestones, or just comparing scheduled dates with actual dates. A formal method called EVA can also be used.

Earned Value Analysis (EVA): EVA can be used to report current and past project performance, as well as predict future project performance. It uses three main measures.

Planned Value (PV)
The portion of the approved cost estimate planned to be spent on the given activity during a given period.

Earned Value (EV)
The value of the work actually completed.

Actual Cost (AC)
The total of the costs incurred in accomplishing work on the activity in a given period. (Eg. how much was actually spent)

Schedule Variance Analysis
Uses EV and PV to calculate the variance to the project schedule.
Schedule Variance: Expressed in dollars

$$SV = EV - PV$$

Schedule Performance Index: Expressed as a fraction

$$SPI = \frac{EV}{PV}$$

Cost Variance Analysis

Uses EV and AC to calculate the variance to the project cost.

Cost Variance: Expressed in dollars

$$CV = EV - AC$$

Cost Performance Index: Expressed as a fraction

$$CPI = \frac{EV}{AC}$$

Agile Planning

Has many differences to formal, the main one being detailed plans are made at the start of an iteration or sprint, and all phases of planning such as requirements, designing and tests are done during a sprint. The requirements are also based on user stories which are light weight and determined by clients. (See planning in scrum slide for more details).

Release Planning

We can generally only pick two out of scope, budget and schedule, which allows us to predict only that value. For example, if scope is fixed, then requirements are stable, but budget or schedule may change.

Fixed Date Release Planning

Used when date is more important, and we have a specific deadline for our project.

- Determine the number of sprints
- Go through the product backlog and estimate and prioritise stories
- Measure the team velocity range (eg. V_{min}, V_{max}), or rate they complete a story
- Compute minimum and maximum story points based on velocity
  $SP_{min,max} = V_{min,max} \times N$
- Draw lines through the product backlog to show the above.

Fixed Scope Planning

Used when scope is more important.

- Groom the product backlog by creating, estimating and prioritising the must have stories
- Determine the number of must have story points (SP_{total})
- Measure the team velocity range (eg. V_{min}, V_{max})
- Compute the min and max number of springs
  $S_{min,max} = \frac{SP_{total}}{V_{min,max}}$
- Show on burndown chart

Cost Estimation (Formal)

Cost Estimation
Estimation of how much money, effort, resources and time that it will take to build a specific software based system or product.

Challenges
No estimation measure can be considered 100% accurate, and no one is able to predict everything that can go wrong in a project. Most estimation methods assume things will proceed as expected, and just adds slack for what may go wrong.

Expert Judgement
A cost estimation technique that uses experts to discuss values until a consensus is reached. The estimated cost is determined by the following equation (p = pessimistic, m = most likely, o = optimistic):

$$e = \frac{(p + 4m + o)}{6}$$

Delphi Technique: An example of Expert Judgement. A panel of experts are asked to give values, and the average is calculated and presented to all of them. Each expert can then choose to revise their estimate after a discussion, and this continues until there are no more changes.

Estimation by Analogy
The cost of a new project is estimated based on similar projects in the same application domain.

Parkinson's Law
This law states the work will expand to fill the time available. The cost is determined by the number of available resources, rather than objectively. For example, if we want software to be delivered in 12 months, and we have 3 people, then the effort is 36 person months. Most of the time this isn't a realistic estimation.

Pricing to win
The cost is estimated to be whatever the customer has available to spend on the project. The cost is dependent on the budget, and other factors like scope and schedule has to be negotiated.

Algorithmic Cost Modelling
A model is developed using historical cost information, based on some software metric (size, code etc) to the project cost. Generally this is done by skilled project managers, who also consult expert for value estimates.

$$Effort = A \times Size^B \times M$$

A = a constant factor that depends on organization practices
Size = Size of software estimated on a metric of choice (lines of code, function points, use case points)
B = Value between 1 and 1.5 which is derived experimentally
M = Multiplier made by combining process, product, and development attributes, such as stability of a requirement or experience of the team.

Steps for calculation:

- Estimate the size of the development product
- Estimate the effort in person-months or person-hours
- Estimate the schedule in calendar months
- Estimate the project cost in agreed currency

Software Size Estimation

Source Lines of Code (SLOC)

Can count physical lines of code (excluding comments and blank lines) or can count logical lines of code, which are the executable statements (which may change depending on language).

Advantages are this method is easy to count and automate, and the size of the software can be measured, eg. more code = bigger system. The disadvantages are that lines of code can vary between languages, and it is very difficult to predict.

Function Points (FP)

Used to express the amount of functionality in a software system. A high number of function points indicates more functionality. Function points can also be used to estimate the cost and effort of the system, predict number of errors and components, and measure productivity. Function points can be computed from software requirements specification.

The advantages is that function points measure the size of the solution rather than the problem, and can be estimated early in the project. It is also independent of technology and programming languages unlike SLOC. The disadvantages are that it relies on well defined requirements at the start of the project, can be time consuming or require experience to estimate well.

Use Case Points

A software estimation technique used to measure the software size with Use Cases. It is intended to be used with UML, and document functional requirements. Use cases model the interaction between actors and a system.
Use cases have several values that need to be computed

1. Compute Unadjusted Use Case Weight (UUCW)
2. Compute Unadjusted Actor Weight (UAW)
3. Compute Technical Complexity Factor (TCF)
4. Compute Environmental Complexity Factor (ECF)
5. Compute the final size estimate

Unadjusted Use Case Weight

Count the number of simple, average of complex use cases N_S, N_A, N_C based on the number of transactions.

Use Case Type	Number of Transactions	Weight
Simple	1-3	5
Average	4-7	10
Complex	8+	15

$$UUCW = N_S \times 5 + N_A \times 10 + N_C \times 15$$

Unadjusted Actor Weight

Count the number of simple, average of complex actors N_S, N_A, N_C.

Actor Class	Actor Type	Weight
Simple	External System using simple API	1
Average	External System using standard protocol	2
Complex	Humans interacting	3

$$UAW = N_S \times 1 + N_A \times 2 + N_C \times 3$$

Technical Complexity Factor (TCF)
Score (S_i) each factor between 0 and 5, where 0 is irrelevant and 5 is essential. Weights (W_i) for each factor are already provided.

$$TF = \sum_{i=1}^{N} S_i W_i$$
$$TCF = \frac{0.6 + TF}{100}$$

Environmental Complexity Factor (ECF) Score (S_i) each factor between 0 and 5, where 0 is irrelevant and 5 is essential. Weights (W_i) for each factor are already provided. Factors are based on team environment, not project complexity.

$$EF = \sum_{i=1}^{N} S_i W_i$$
$$ECF = 1.4 + (-0.03 \times EF)$$

Final Size Estimate
Combines all the factors above.

$$UCP = (UUCW + UAW) \times TCF \times ECF$$

The advantages of UCP are that they can be measured early in the project, and estimates are accurate if done by experienced people. They are also easy to calculate. The disadvantages are that requirements have to be written in the form of use cases, and technical and environmental factors have a high impact on UCP.

COCOMO Model

Derived from collecting data from a large number of software project and deriving a formulae that best fits the observation. It is an empirical model, and is completed in the design phase of planning.

$$Effort = A \times Size^B \times M$$

We recall how to calculate Size, and assume we use KSLOC as our size measurement. Assuming we have 200 Function points, with an average of 148 lines of code in C, the KSLOC is calculated below.

$$KSLOC = \frac{200x148}{1000} = 29.6$$

Estimating parameter B, we use the following equation:

$$B = 1.01 + 0.01 \sum_{i=1}^{5} W_i$$

Where the weight is a scaling factor ranging from 0 -5.

Finding M, we use a number of factors, rated on low to extra high values which are precalculated.

$$M = RCPX \times RUSE \times PDIF \times PREX \times PERS \times SCED \times FCIL$$

From the effort calculated, we can estimate time and number of personnel (N).

$$T = 2.5 \times Effort^{0.33+0.2(B-1.01)}$$

$$N = \frac{Effort}{T}$$

Cost Estimation (Agile)

Story Points: A relative measure of the size of a user story (compared to other user stories).

Velocity: A measure of productivity of the team, can be calculated by the number of story points delivered in a specified time period.

Process

- Develop user stories for the system
- Estimate the number of story points for each story using a chosen technique
- Use the teams velocity from previous experience to estimate the delivery time of the project. Develop a burn down chart if scope is fixed.
- Measure the actual velocity of the team during development.
- Using this velocity, re-estimate the time it will take to deliver the product.

Agile Estimation Guidelines

Estimate by analogy
There are no units for story points, so the number of story points is always relative to other story points. For example, A = B = 2C.

Decompose a story
We can decompose a story into smaller tasks, and use them to provide a total measure. Eg. If a task is too complex, break it up then compare to other story points.

Use the right units
Relative units are generally done as 1,2,4,8 etc, rather than being too fine. It's much easier to judge stories as double the effort than 0.123.

Use group based estimation
The group is generally involved in the story points assigning process. Techniques such as the delphi method can be used to reach a consensus.

Planning Poker

- Customer reads a story
- Team estimates
- Team discusses
- Team estimates again until a consensus

Bucket System

- A user story is picked and put in a bucket
- The next few cards are picked randomly, and discussed with the team to be placed in a bucket relative to the previous ones
- The rest of the cards is divided up into the team, who are responsible for assigning the rest of the cards
- The team discusses the placement of the cards and reaches an agreement.

Relative Mass Valuation

- A story is picked, and the team rates it as large, medium or small
- The large stories are at one end and smaller ones are at the other end
- The team continues picking, discussing and separating stories based on size and positioned in order of smallest to largest.
- The team assign points to the easiest story, and moves up the list until we reach a story that is assigned a 2, 4 etc.

Velocity

$$V = \frac{SP}{T_i}$$

Where SP = story points completed, and T_i = time period that they were completed.

Estimated Delivery Time

$$T = \frac{\sum_{i=1}^{N} SP_i}{V}$$

Teams and Motivation

Motivation

As a manager it is important to know how to motivate those around you. Particularly in an Agile environment, you want to be able to create an autonomous team that is able to work together without a manager explicitly telling them what to do.

Organisation Theory

There are various approaches to organisational analysis. Organisations are groups of individuals that has a structure, and are managed to meet a need or achieve a collective goal.

Maslow Hierarchy of Needs

An old model that focuses on tiers of human needs. Essentially the basic needs comprise of the bottom most tiers, and need to be met before higher needs. For example, access to food and safety is a low tier need, and promotions and confidence would be a high tier need. While this could be outdated, as you can have multiple needs of different priorities at once, fundamentally issues like a living wage need to be addressed before other aspects like fostering a creative environment.

Hertzberg Two Factor Theory

There are two sets of factors (determined empirically) that cause job satisfaction and job dissatisfaction. To remedy causes of dissatisfaction will not create job satisfaction and vice versa. If a PM wanted to improve one, they would look at the factors for it, eg. Job satisfaction is related to personal growth, so maybe the PM can recommend a new position. Note that the PM won't always be able to control these factors directly (like salary).

Leadership

While PMs are paid to manage a project, they also need to be able to influence others and lead them. Managers can set deadlines and assign work, but leading them involves motivating others and inspiring them to achieve a goal.

Common Issues

Some common assumptions PMs can make is that all people have the same motivations, whether that is the same as them or money. Realistically everyone has their own personal motivations, and PMs should learn more about individuals to know how to motivate them. Another assumption is that people don't need motivating or only need to be motivated in certain situations. Different leadership styles may be needed to suit different teams and projects and this should always be a consideration when managing a team.

Teams

A team is two or more individuals working together to achieve a common objective. A group is a collection of people who work together but don't necessary have the same goals. A group becomes a team when all members demonstrate a commitment to each other and a common goal.

Advantages of Teams

- Very few individuals possess all the knowledge, skills and abilities to accomplish all tasks
- Complementary teamwork skills are one of the most commonly required skills in the work environment
- Substantial benefits to the organisation and team members, allow division of work, faster progress
- Shard accountability increases the likelihood of success

Team Formation

The Tuckman's Team Development Model has four stages; Forming, Storming, Norming and Performing. An Adjourning stage was later added to the model.

Forming: Team establishes ground rules and formalities. There is a high dependency on the leader.
Storming: Members communicate but maintain strict individuality. The leader is there to coach and support.
Norming: Team bonding, and higher acceptance of perspectives. The leader becomes a facilitator and enables this bonding.
Performing: There is less emphasis on the hierarchy and more on the flexibility.
Adjourning: Yearly assessment and plan for acknowledging individual contributions. The leader acknowledges, recognises and directs.

Effectiveness of a Team

Signs of a good team would be clear communication and participation from all team members. Regular meetings and signs that a team is working together rather than individually. A sign of a bad team would be lack of communication, working individually all the time, and not supporting others or blaming them for issues.

Team Structures

Controlled Centralised

The leader coordinates tasks and directs work. The communication and control structure flows downwards, eg. Leader → Coordinator → Teams. Sub teams can have leaders to direct and guide subgroups.

Scrum Team

The Scrum Master doesn't have a hierarchy in a scrum team. While they are a leader and there for technical advice, the scrum team is on equal footing, and the relationship is a servant leadership, where they exist to help the members of the team. The communication flow could be Business Owner → Product Owner → Scrum Team.

Leadership Styles

Authoritarian or Autocratic

- Rarely consults team members, and is autonomous in decision making
- Good for a crisis or when a task needs to be completed quickly
- Can be demoralising, and demotivate people with constant use
- Can also miss out on other ideas and perspectives, and stifles innovation and creativity

Democratic or Participative

- Actively involves team members in the process and builds trust
- Not great in a high pressure environment as it can be a slow process
- Not good with handling disagreement or conflict, as others may not respond well to consultation. Might need an autocratic approach.

Delegating

- Team members have free rein on how they work
- Ideal when working with people who are highly skilled and motivated, or you work with people you trust
- If a member is inexperienced or untrustworthy, could have disastrous consequences
- Though everyone is involved in decisions, as a leader you will be responsible for the outcome

Quality Management

End user's Perspective
End users judge the quality of a product by their interaction with it. Generally a system should be reliable, have reasonable performance be easy to learn and use. If the functionality is hard to learn but is extremely important then users may still consider it to be a high quality system. These are considered **external quality characteristics**, as they are typically associated with the external behaviour of the system.

Developer's Perspective
The developer's perspective typically also includes the number of faults that the system has, the ease of modifying the system, ease of testing the system, ease of understanding the system design, re-usability of components, conformance to requirements, resource usage and performance. These are mainly **internal quality characteristics**, because they are concerned with the quality of the internal structure of the system.

Quality of Cost
If quality is factored into the requirements, it would be a small cost to fix. If found after deployment or in a later stage, it could be very expensive to fix.

Quality Assurance
The establishment of a framework of organizational procedures and standards that lead to high-quality software. Involves the definition or selection of quality standards.

Quality Planning
The selection of appropriate procedures and standards from the framework, adopted for the specific project. Intended outcome of this process is a Software Quality Plan (SQP or Software Quality Assurance Plan))

Quality Control
Ensuring that the software development team has followed the project quality procedures and standards.

Verification
An attempt to ensure that the output of the product matches the requirements of the project. Eg. for a find the cheapest product algorithm, it ensures that the output is a product (though it may not be the cheapest). Verification usually involves the requirements specification vs design, and design vs code, which is done internally.

Validation
An attempt to ensure that the product achieves its purpose. Eg. for the find the cheapest product algorithm, the product that it outputs is the *cheapest* product. Usually this involves going back to the stakeholders to check that the product meets their requirements, which is usually done externally.

Product Standards
Apply to the product being developed. Eg. Coding standards to follow, documentation standards, requirements document structure.

Process Standards
These standards define the processes that should be followed during software development. Eg. Design review, validation, version control etc.

Standards Advantages

- Provide a framework around which the quality assurance process may be implemented
- Provide encapsulation of best, or appropriate practices
- Customers may also require a particular quality standard or level when choosing a software vendor

Standards Disadvantages

- Not seen as relevant or up to date by software engineers
- Involves too much bureaucratic form filling
- Unsupported by software tools so tedious manual work is involved ot maintain standards

Capability Maturity Model

1. **Level 1: Initial** - Processes are predicable, poorly controlled and reactive. No prior plans in place for projects.
2. **Level 2: Managed** - Processes are characterised for projects, but still often reactive. Has some processes in place, but not extensive coverage of all situations.
3. **Level 3: Defined** - Processes characterized for the organization and is proactive. Projects also tailor their processes from organization's standards. They have processes in place, and follow a procedure.
4. **Level 4: Quantitatively Managed** - Processes measured and controlled. Has quality management, risk control etc.
5. **Level 5: Optimizing** - Focuses on improving processes, always ensuring that the processes created are up to date and relevant according to business and organizational guidelines.

Software Quality Assurance Plan

- Product Overview: A description of the product, intended market, and quality expectations.
- Product Plan: The critical release dates and responsibilities, could point to the schedule.
- Quality Goals: The quality goals and plans for the product, including identification and justification of critical product quality attributes.
- Process Description: The quality assurance processes that should be used for product development and management (reviews, audits etc)
- Document and Coding Standards: Standards for documents and coding standards
- Risks and Risk Management: The key risks that might affect product quality, and the actions to address these risks (Risk Management Plan)

Technical Reviews

Reviews of artefacts is performed by peers in the development team but the authors are involved. The aim is to uncover problems in an artefact and seek ways to improve it. Eg. Someone else looking or testing code.

- Informal Reviews: Simple desk check or meeting with a colleague. Very casual, and aims to check off standards.
- Formal Reviews: A meeting with multiple stakeholders or a review team. Involves recording comments, a review leader, author, reviewers and recorder. This team can accept, modify or reject the artefact.
- Walkthroughs: An author leads a group of reviewers through the code, and where needed is given suggestions or solutions.
- Code Inspections: Similar to formal reviews, with a focus on code.
- Audits: Similar to a formal review, without an author. Usually done by a team external to the organization, and they can check the product or processes in a project.

Business Reviews

Ensures that the IT solution provides the functionality specified in the project scope and requirements document. Can ensure that the project is complete, provides information needed for the next phase or process, and meets all the standards.

Management Reviews

Compares the project's actual progress against a baseline project plan. The PM is responsible for presenting progress and status update, may need changes to project if issues occur. May also involve reviewing if the project meets the scope, schedule, budget and quality objectives.

Contracts, Outsourcing and Procurement

Outsourcing
The practice of engaging an external party (under contract) to perform services or create goods that are traditionally performed in-house by the company's own employees.

There are multiple types of outsourcing:

- **Onshoring:** Assigning to another party inside national borders to access targeted benefits. (i.e within Australia)
- **Nearshoring:** Assigning to another country within close proximity (eg. New Zealand)
- **Offshoring:** Assigning to another country irrelevant of geographical location or timezones (eg. China, US, Europe)

Pros to outsourcing:

- Cheaper
- Can access specific skills (if uncommon)
- Saves time or resources for the business
- Access to better technology, or best practice (eg. specialised businesses)

Cons to outsourcing:

- Loss of control
- Less communication
- Security issues
- Management over parties
- Differences in quality, expectations, rules

Procurement
The Procurement Management Process is used when outsourcing is required in a project. It consists of three main stages, **Plan**, **Source** and **Manage**.

Planning
Involves consulting key stakeholders to define the 'real' need, analysing how the supply market works, assessing risks and ultimately defining the best Procurement Strategy to meet the organisations requirements.

P1. Analysing Business Needs
Determining the areas where outsourcing is required, and what benefit it will bring to the business. Eg. If a task needs to be outsourced, or done internally.

P2. Analyse and Engage Market
Research the costs, suppliers and options that are currently in the market.

P3. Finalize Procurement Strategy
Define the best Procurement Strategy to meet the requirements. For examples, considering the risks, making a plan and determining which strategy the organization would like to proceed with. (Extended in the next step)

Sourcing
The principle objective of this stage is to identify and engage suppliers who will provide the best value for money outcome, in a framework of probity and fair dealing. A key deliverable in this stage is to determine the appropriate sourcing method, with consideration given to alternatives other than just tendering.

S1. Approach the Market
Engage suppliers, provide them with an invitation to bid, quote etc. Gathers all the necessary information required to make a decision.

S2. Select
Evaluates all the information gathered, and assigns a ranking or weight to each one based on observed merits or needs.

S3. Negotiate and Award
After the supplier is selected, we then negotiate the terms of the contract before awarding it. This is done with both parties, and discusses topics such as costs, schedule, quality etc.

Sourcing Procurements

The procurement process is typically conducted with the issuing of a Request For x (RFx) where x = Bid, Information, Proposal, Tender or Quote.

- **RFB (Request for Bid):** This is an invitation for prospective suppliers to bid on service, it's more of an expression of interest rather than a binding agreement.
- **RFI (Request for Info):** This gathers information for potential suppliers. Usually done before a proposal or offer.
- **RFP (Request for Proposal):** A document is posted to elicit bids from potential vendors. It specifies an evaluation criteria and is used for complex IT projects or to boost competition.
- **RFQ (Request for Quotation):** A document eliciting quotes for a product or service. It's main purpose is to seek an itemized list of prices, and is used for simpler IT projects.
- **RFT (Request for Tender):** Invitation for suppliers to submit a sealed bid. It specifies the services required, and the timeframe for the project. This is usually expected to conform to a legally standardized structure.

The RFx document is prepared by the buyer and will have specific information that depends on the type of document. It usually includes: A purpose, organization's background, basic requirements for the project, hardware and software environments, a description of RFx Processes and Evaluation, a Statement of Work and Scheduled information, as well as relevant appendixes such as a sample contract or system requirements.

Statement of Work

A key component of the RFx document, which is a description of the work required. It should be detailed and give bidders an understanding of the buyer's expectations. It should include the following:

- Scope of Work to be completed
- Location of where the Work is to be completed from
- Measurement and Performance criteria
- Deliverables, milestones and schedule
- Applicable Standards and Acceptable Criteria
- Any special requirements

Evaluation

After all the suppliers have submitted their information, the next step would be to evaluate their merits with an evaluation team. Usually this team would have some predetermined criteria for ranking. From this, they can short-list suppliers for a presentation, check their references and select short-listed firms. They will then give a best and final offer (BAFO) with selected firms, then depending on the outcome, select a supplier with the best terms and conduct the final negotiations.

Manage

After a contract with the vendor has been confirmed and awarded, the next step would be to implement, manage and review. This is an ongoing process for the rest of the project which involves managing their work for status updates and ensuring their project is on track, reviewing any changes made and then finally ensuring that the final product is up to standards and they have access to all the information required for any future development.

M1. Implement Arrangement

Implements the agreement and services as per the contact and SOW. Establish frequency and methods of communication and project updates, as well as any other items that should be discussed prior to the commencement of the project.

M2. Manage Arrangement

An ongoing process, often done by the project manager. Ensure that updates about the projects are provided as necessary and communicated to stakeholders in the project updates (eg. updated about PM's team and outsourced team). Ensure that the seller's performance is meeting all contractual requirements (on schedule for milestones etc). Any changes also need to be reviewed and controlled by the PM.

M3. Renew

Involves completing, settling contracts and resolving issues. The project team needs to determine if all the work was completed correctly and to the required standards (mentioned in contract), resolve any issues or outstanding items. They should also update records to capture all lessons learnt and outcomes using this supplier (for future reference), as well as information required for what was outsourced (in case they have a question or want to develop it further). The contract should also have an outline of the requirements for this process.

Contracts

A mutually binding agreement that obligates the seller to provide the specified products or services, and obligates the buyer to pay for them. It clarifies the responsibilities of the seller, and focuses on key requirements such as deliverables, quality and timeframes. It should be detailed and accurate and contain the final position of both parties, can't really change without paying more, and should ensure that quality, budget or timeframes are met.

Fixed Price Contract

Involves a fixed total price for a well-defined product or service.

Has a low risk for buyer, high risk for seller. Usually chosen if the deadlines are clear, there is a detailed specification, short project duration and no changes planned.

Pros: It is low risk for the buyer, they won't go over budget, and don't have to constantly supervise the seller for updates. They both agree on a product and price, and at the end of the contract, the buyer should receive the agreed upon product.
Cons: That the buyer has less control over the process and less communication with the seller. Additionally, more upfront work is required (such as setting requirements and planning deadlines).

Time and Material Contract

Involves payment to a seller for the actual time spent and any materials used in providing the service.

Has a high risk for buyer, but low risk for seller. (Buyer won't know how long the project will take, and might not have everything finished as intended). Used when there is only a raw project concept, the workflow can change, or more requirements can be added, if little is known about the target market, or if there is the intention of the buyer taking control of the project.

Pros: A flexible budget (seller gets paid what they worked for), easy to start as not as much planning is needed, part-payment opportunity, no costs for preparations, and usually suits an agile orientation.
Cons: No deadlines, buyer won't know when it will finish, low budget control, won't know how much they will need to spend, time for participation, buyer will likely be controlling the project and wanting status updates etc.

Contracts

Important items to include in contracts are specific clauses that take into account issues that are unique to the project, such as quality, time, location etc.

- Intellectual Property Ownership and Indemnities (who will own the code after it is made)
- Milestones and Deliverables
- Quality Criteria, Performances, and Acceptance Testing
- Variations or change request process
- Contract Termination, Non-performance etc.
- Disengagement and Transitions
- Liquidated Damages
- Fees and Penalties
- Warranties

Ethics

Organizational Ethics
Express the values of an organization to its employees and other entities irrespective of governmental and regulatory laws.

Individual Ethics
The principles and values used by an individual to govern his or her actions and decisions.

Importance
Ethics in organizations are important as they satisfy basic human needs, being fair, honest and ethical are important to everyone, and each employee should feel that they work in an organization that has fair and ethical practices. They also create credibility for society, unite people by having similar beliefs, and setting the basis for decision making.

Australian Computer Society Code of Ethics

- Primacy of Public Interest
- Enhancement of Quality of Life
- Honest
- Competence
- Professional Development
- Professionalism

Configuration Management

Software Configuration
Software projects generate a large number of different types of artefacts, where some may have dependencies. Software configuration is the total of all the artefacts, their current state and the dependencies between them.

Configuration Management
Used to manage how to make changes to artefacts, which may impact all the dependencies. It should manage change properly without losing overall consistency by establishing processes, setting up repositories for version control, and using other tools as necessary.

CM Aims

- To identify all items that collectively will make up the configuration
- To manage changes to one or more of these items so that the collection remains consistent
- To manage different versions of the product
- To assure software quality as the configuration evolves over time

CM Tasks

- **Identification:** The configuration items necessary for the project are identified, as well as any dependencies
- **Version Control:** Processes and tools are chosen to manage different versions of configuration items as they are developed
- **Change Control:** Changes that affect more than just one configuration item are managed
- **Configuration Auditing:** The consistency of the configuration is checked
- **Configuration Reporting:** The status of configuration items is reported

Identification
The set of artefacts that require configuration management are called the configuration items. They can be **Basic**, **Aggregate** or **Derived**. Basic is a single configuration item, Aggregate can consist of many configuration items and Derived is the product of other configuration items. A typical list of configuration items can be requirement specifications, use cases, design documents, source code, release modules, test script etc.

Version Control

- A repository for storing configuration items
- A version management function that allows software engineers to create and track versions, and roll the system back to previous versions if necessary
- A make-like facility that allows engineers to collect all configuration objects and build it

Version
An instance of a model, document, code or other configuration item which is functionally distinct in some way from other system instances.

Variant
An instance of a system which is functionally identical but non-functionally distinct from other instances of a system.

Release
An instance of a system which is distributed to users outside of the development team.

Derivation History
This is a record of changes applied to a configuration object. Each change should record the change made, the rationale for the change, who made the change, when it was implemented. A common method of tracking versions is through version numbering (1.1, 1.2, ... 2.0)

Version Control
Tracks changes to the code or documentation, is done during development and made to ensure that multiple people can make changes to the same project.

Change Control
Changes that affect more than just one configuration items are managed. Usually requires approval from stakeholders or PM before it is put in production. Also need to consider benefit and risks of changes.

Change Management Plan
Allows changes to the configuration and ensures it is done in a way that allows everyone to be aware of what needs to be done, how it would affect them and reasoning for the changes. Essentially, initiates the change, evaluates the change then makes the change.

Baseline
An artefact that is stable, it has been formally reviewed and agreed upon and is now ready for future development. It can only be changed through a formal change management procedure.

Configuration Audits
Complements the other configuration management activities by assuring what is in the repository is consistent, and all the changes have been made properly. Also checks other processes such as if the changes were approve, the quality of changes etc.

Status Reporting
A common way for large projects to keep track of status of the repository, the idea is to review the configuration objects for consistency with other objects, and can find other omissions or potential side effects. Usually the aim is to report on the status of the configuration items of interest and the baselines that have been achieved.